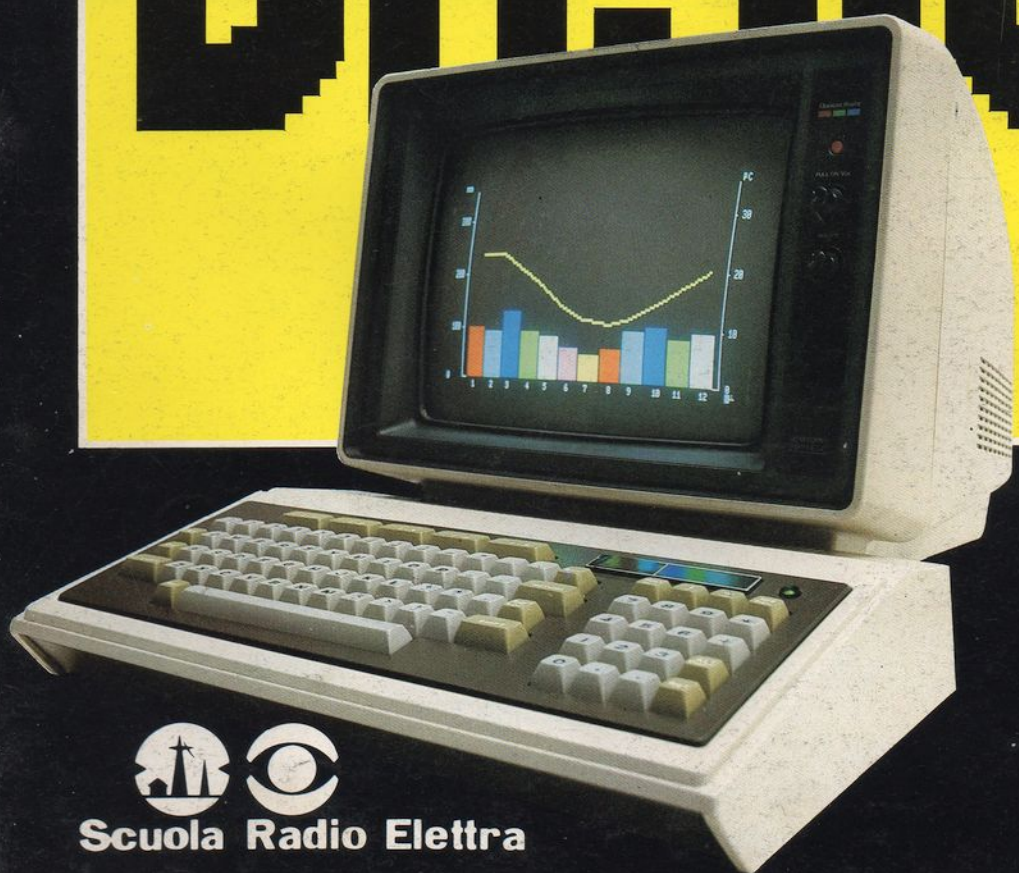
PARLA BASIC



Scuola Radio Elettra

Scuola Radio Elettra
Torino

Analisi e programmazione

Lezione 8

1 — Organizzazione delle informazioni
2 — Strutture concrete dei dati
3 — Organizzazione dei dati nella memoria principale
4 — Organizzazione dei dati nella memoria ausiliaria
5 — Cenni sull'organizzazione delle informazioni nei grandi sistemi di elaborazione

Che cosa apprenderà

Nella presente lezione verrà esaminata l'organizzazione delle informazioni da elaborare nella memoria principale e nella memoria ausiliaria o di massa del computer: nastro magnetico e disco.

Sarà, inoltre, ampliato il concetto di FILE, insieme di dati di qualsiasi tipo, e saranno definiti i suoi sottoraggruppamenti: RECORD e CAMPO.

Completerà la lezione un breve esame dell'organizzazione delle informazioni nei grandi sistemi di elaborazione.

Pagina mancante

e la tabella di informazioni seguente.

VENDITE CLIENTI		
Codice	**Descrizione**	**Importo**
1001A	ROSSI G.	500.000
1002A	FERRERO M.	1.000.000
1003A	ABATE F.	1.500.000
1004A	MONTI V.	2.500.000
1005A	BELLINI F.	600.000
1006A	MONTALTO A.	3.450.000
1007A	ROSSINI P.	4.000.000
1008B	MARCONI P.	6.000.000
1009B	CALOGERO P.	7.000.000

Il criterio di organizzazione degli elementi di queste strutture è puramente logico, cioè non dipende dal particolare supporto fisico su cui essi sono raffigurati (nel nostro caso tale supporto è costituito dal piano del foglio di carta).

Per vedere se le strutture presentate sono puramente logiche, basterebbe provare a organizzarle in qualche altro modo e controllare se l'aggregazione dei vari elementi resta "invariata".

Per esempio, la matrice:

B (2, 3, 3) a tre dimensioni

potrebbe essere rappresentata come mostrato nella *fig. 1* tridimensionale.

Come si vede, ad ogni elemento b_{ijk} della matrice viene fatto corrispondere un cubetto elementare, posto in un punto dello spazio a tre dimensioni di coordinate:

P (i, j, k).

Gli elementi b_{ijk} rappresentati nella *fig. 1* sono organizzati in tre "cataste" di sei elementi ciascuna, corrispondenti alle tre colonne della stessa matrice B (2, 3, 3) raffigurata precedentemente su un piano a due dimensioni.

Se si cambia opportunamente il "punto di vista", si constata che le due strutture organizzate di dati corrispondono e si identificano nella stessa

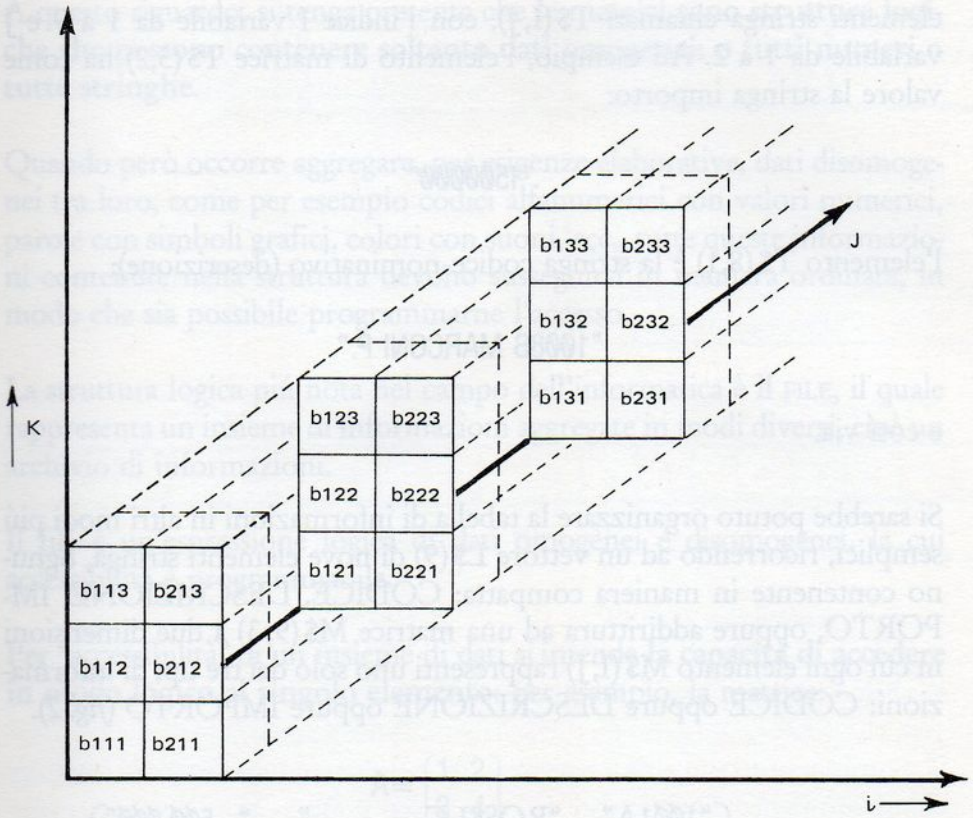


Fig. 1 – Rappresentazione astratta di una matrice B(2, 3, 3) a tre dimensioni.

matrice B(2, 3, 3), anche se si utilizzano per la loro rappresentazione due diversi supporti fisici: il piano a due dimensioni per la prima struttura e lo spazio a tre dimensioni per la seconda struttura.

Analogamente, la tabella di informazioni riportata in precedenza si può organizzare logicamente in una matrice stringa a due dimensioni T$(9,2) cosí definita:

$$T\$(9,2) = \begin{pmatrix} \text{“1ØØ1A ROSSI G.} & \text{”} & \text{“ 5ØØ.ØØØ”} \\ \text{“1ØØ2A FERRERO M.} & \text{”} & \text{“1.ØØØ.ØØØ”} \\ \text{“1ØØ3A ABATE F.} & \text{”} & \text{“1.5ØØ.ØØØ”} \\ \text{“1ØØ4A MONTI V.} & \text{”} & \text{“2.5ØØ.ØØØ”} \\ \text{“1ØØ5A BELLINI F.} & \text{”} & \text{“ 6ØØ.ØØØ”} \\ \text{“1ØØ6A MONTALTO A.} & \text{”} & \text{“3.45Ø.ØØØ”} \\ \text{“1ØØ7A ROSSINI P.} & \text{”} & \text{“4.ØØØ.ØØØ”} \\ \text{“1ØØ8B MARCONI P.} & \text{”} & \text{“6.ØØØ.ØØØ”} \\ \text{“1ØØ9B CALOGERO P.} & \text{”} & \text{“7.ØØØ.ØØØ”} \end{pmatrix}$$

Questa matrice è costituita da nove righe × due colonne, cioè da diciotto

Pagine 6-15 mancanti

Pagine 6-15 mancanti

ziale; **tutti i dati,** cioè, **vengono disposti in successione, uno dietro l'altro,** ed il supporto fisico magnetico del nastro presenta, ad intervalli regolari, delle zone "vuote", ossia spazi non incisi, detti GAP (*fig. 8*).

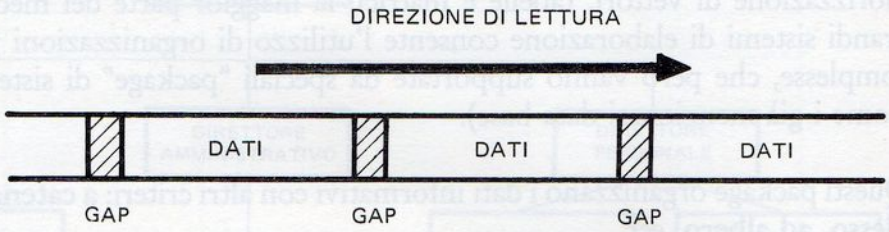


Fig. 8 – Rappresentazione schematica di un "file sequenziale" su nastro magnetico.

I gap si formano a causa della inerzia meccanica del sistema di registrazione, che, durante le operazioni di lettura/scrittura di dati, ha bisogno di un intervallo breve di tempo e di spazio per portarsi alla prevista velocità di scorrimento del nastro magnetico.

Le zone di nastro magnetico indicate con dati sulla *fig. 8* sono utilizzabili per la registrazione di informazioni (numeri e parole), generalmente organizzate in gruppi e sottogruppi (record, campi, sottocampi) disposti in sequenza secondo lo svolgimento della bobina di nastro (file sequenziale).

Durante il trasferimento di dati fra il sistema di memorizzazione esterno e l'elaboratore, è necessario spostare un intero record.

Alcuni sistemi fanno uso di dimensioni fisse per i record, il che significa che tutti i record hanno la medesima lunghezza.

La maggior parte dei sistemi a nastro, tuttavia, prevede l'impiego di record di lunghezza variabile, per cui un record può avere una lunghezza di valore desiderato.

Occorre però tenere presente che l'impiego di record di lunghezza breve può comportare una diminuzione notevole della capacità di memorizzazione, a causa della presenza degli intervalli che sussistono fra record adiacenti.

La velocità di trasferimento (in inglese, "transfer-rate") di un sistema, espressa in byte al secondo, costituisce una misura della velocità con la quale i dati possono venire letti oppure scritti nella memoria ausiliaria.

Nei sistemi a nastro magnetico le velocità di trasferimento sono comprese fra circa 25 byte al secondo per registrazioni audio normali ed oltre 1.000 byte al secondo per registrazioni digitali di alta qualità.

Il nastro magnetico, specialmente quello montato in cassetta, è senz'altro il mezzo piú economico e diffuso per la realizzazione di memorie ausiliarie.

Il reperimento di un certo record depositato sul nastro e la lettura dei dati in esso contenuti, cioè il travaso di questi nella memoria del computer, costituiscono un'operazione fondamentale.

Se la posizione del record non è conosciuta, la soluzione migliore consiste nel riavvolgere il nastro e nell'effettuarne la lettura fino a che il record desiderato non viene individuato e trasferito nella memoria principale.

Questa operazione, naturalmente, può richiedere diversi minuti, anche con un sistema basato su cassette numeriche ad elevate prestazioni.

4.2 - La registrazione dei dati su dischi

Anche per i dischi si utilizza la stessa struttura organizzativa dei nastri: file, record, campi, sottocampi.

I file, però, oltre che essere disposti sequenzialmente sul supporto magnetico del disco, possono essere organizzati in un altro modo (o in piú modi diversi, se si opera con sistemi di elaborazione piú complessi).

Struttura RANDOM

E' ciò che viene definito **struttura RANDOM** o ad **accesso diretto**, definizioni equivalenti, che in sostanza stanno ad indicare lo stesso criterio di memorizzazione su disco.

Le informazioni registrate su nastro, raggruppate sia in campo sia in record, sono sempre organizzate in sequenza; per questo motivo l'accessibilità ad un dato specifico è lunga, perché presuppone il passaggio di tutte le informazioni che lo precedono.

Le informazioni registrate su disco, al contrario, sono accessibili piú rapidamente, perché la ricerca di un dato specifico non implica necessariamente il passaggio di tutte le informazioni che "fisicamente" lo precedono.

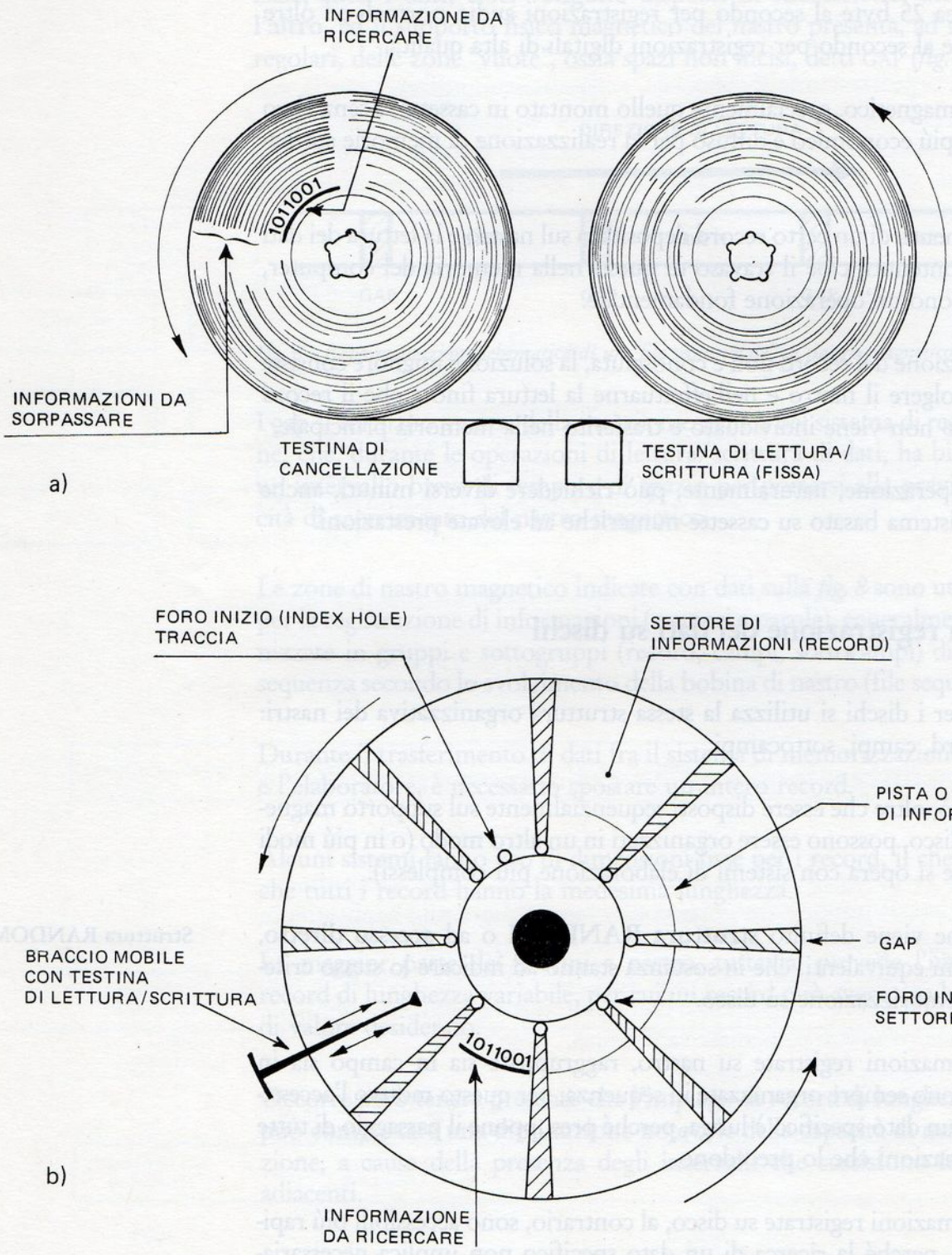


Fig. 9 – Confronto tra memoria a nastro (a) e memoria a disco (b).

Nella *fig. 9* è riportato uno schema grafico che illustra il criterio di ricerca di una specifica informazione sia su nastro magnetico sia su disco, piú precisamente su un floppy-disk usato nei personal computer. Occorre precisare che per qualsiasi tipo di disco, rigido o flessibile, **l'organizzazione dei dati è praticamente sempre la medesima.**

Come si può constatare, il nastro è un supporto di memoria che non consente rapidi spostamenti e "salti" diretti alla specifica informazione; il disco, invece, essendo configurato in piste concentriche (a loro volta suddivise in settori) e potendo ruotare intorno ad un asse di rotazione, riduce notevolmente il tempo fisico di posizionamento della testina di lettura/scrittura sulla specifica informazione.

Osservando la *fig. 9-b*, si può rilevare che le **tracce** o **piste** sono i "solchi" del disco nei quali i bit sono disposti uno di seguito all'altro, mentre i **settori** sono le porzioni di piste delimitate da due raggi del disco e rappresentano la massima lunghezza consentita del record.

L'identificazione dei vari settori del disco viene ottenuta mediante due sistemi.

Il primo sistema, denominato HARD-SECTORED (settorizzazione hardware), consiste nel contrassegnare fisicamente i diversi settori mediante un piccolo foro praticato sul disco stesso, come mostrato nella *fig. 9-b*.

Un ulteriore foro (INDEX HOLE) consente di individuare l'inizio della traccia. I fori in questione vengono rilevati mediante un sistema optoelettronico, disposto nel drive.

I dischi hard-sectored non devono essere confusi con gli "hard-dischi", che sono invece dischi rigidi.

Il secondo sistema di identificazione dei settori del disco, denominato SOFT-SECTORED (settorizzazione software), prevede un solo piccolo foro sul disco (index hole) in prossimità del foro centrale che serve da riferimento per l'individuazione della traccia. In questo caso i vari settori del disco non vengono contrassegnati fisicamente, ma con apposite informazioni registrate sulla traccia mediante un'operazione preliminare, denominata FORMATTAZIONE.

I dischi hard-sectored e soft-sectored non sono fra loro compatibili.

Verso il centro del disco, sia esso del tipo hard-sectored sia del tipo

soft-sectored, vi è una apposita traccia, denominata DIRECTORY, nella quale si trova l'indice del contenuto del disco stesso ed il modo di ritrovare questo indice per caricare un programma oppure dei dati.

La velocità di trasferimento dei dati nei sistemi a floppy-disk è compresa fra 1.000 e 20.000 byte al secondo.

Attualmente stanno sempre piú diffondendosi nei personal computer i dischi rigidi (hard-disk), i quali hanno velocità di trasferimento dei dati piú elevate, normalmente comprese fra 10 Kbyte e 1.000 Kbyte al secondo.

Da quanto sinora precisato si può rilevare che la velocità di trasferimento dei dati dei dispositivi a disco è sensibilmente piú elevata di quella dei sistemi a nastro magnetico; quest'ultimo, pur se poco adatto per archiviare informazioni (le quali devono rendersi prontamente disponibili), consente però di memorizzare una mole di dati piuttosto notevole e può quindi considerarsi un supporto molto utile per la conservazione di dati storici.

Il disco, invece, è utilizzato piú propriamente per l'archiviazione di dati di pronta disponibilità (quindi accessibili e ricercabili velocemente).

Il D.O.S.

La maggior parte della registrazione su disco viene eseguita mediante una serie di procedure denominate D.O.S. (Disk Operating System, ossia "sistema operativo a dischi") del sistema, procedure che vengono generalmente fornite dal costruttore, data la loro complessità.

Nell'interno di ogni disco, in determinate tracce o piste, sono registrate particolari informazioni di carattere organizzativo, relative ad ogni file o archivio di dati registrati sul disco stesso.

Queste informazioni, che vengono utilizzate dal sistema operativo per una veloce ricerca dei dati, riguardano la dislocazione fisica degli stessi dati sulle tracce del disco, dislocazione individuata da due parametri: **numero di pista o traccia** e **numero di settore**.

Ogni tipo di disco ha un determinato numero di piste (nel caso dei floppy-disk da 5 pollici le piste sono normalmente 35) ed un determinato numero di settori (generalmente 16) per ciascuna pista.

Le memorie ausiliarie per i grandi sistemi

Dischi di maggiori dimensioni, di solito utilizzati in grandi sistemi di elaborazioni, hanno un numero di piste notevolmente superiore (circa

un migliaio), ed ogni pista è suddivisa in un numero proporzionato di settori.

Quando da un programma parte l'ordine di ricercare una specifica informazione registrata in un determinato record dell'archivio memorizzato su disco, il sistema operativo del computer provvede ad "interrogare" la speciale pista di informazioni organizzative registrata sul disco: in questa pista, ad ogni record del file o archivio sono associati il numero di pista ed il numero di settore in cui il record è registrato su disco.

Il sistema operativo, attraverso la conoscenza di questi due parametri, esegue quindi un rapido posizionamento della testina di lettura/scrittura sulla zona di disco in cui è registrato il record che interessa.

Dopo essere stato rintracciato, quest'ultimo viene letto (o immesso) nella memoria principale mediante un apposito comando BASIC del programma.

Una volta che il record è in memoria, altre istruzioni di programma provvederanno ad effettuare una ricerca piú minuziosa, fino ad individuare l'elemento di informazione specifico che occorre trovare.

Il record si può considerare un sottoarchivio particolare, creato per evitare che in un'operazione di lettura l'intero file di informazioni possa, per cosí dire, "invadere" tutta la memoria del computer, cancellando eventualmente una parte del programma.

Esso può essere suddiviso, come già precisato, in altre parti chiamate campi, che a loro volta possono essere ripartiti in informazioni ancora piú elementari.

Il record è definito anche **unità minima di trattamento**, cioè un insieme di caratteri che tutto in una volta può essere immesso in memoria o registrato su disco.

Nei sistemi personal computer, dotati di floppy-disk a 16 settori e 35 piste, la lunghezza di un record di solito viene fatta coincidere con la lunghezza di un settore di disco (che è di 256 byte).

Sono anche reperibili in commercio floppy-disk con 40 e 77 piste o tracce. Naturalmente, quanto piú numerose sono le tracce, tanto maggiore è la capacità di memoria del disco. Inoltre, si possono avere dischi **a singola faccia** di registrazione ed **a doppia faccia** di registrazione: questi

ultimi permettono ovviamente una memorizzazione doppia di dati.

Nella transizione di informazioni dal disco alla memoria e viceversa, le informazioni passano in apposite zone della memoria principale chiamate BUFFER (si pronuncia *baffer*).

Il buffer è quindi una zona temporanea di stazionamento della memoria principale attraverso cui transitano le informazioni che dalla memoria devono essere inviate al disco e viceversa.

La capacità di un buffer, espressa in byte, è normalmente uguale a quella di un record di disco.

Per esempio, nei personal computer piú comuni un buffer di memoria conta 512 byte, dei quali 256 sono riservati alle informazioni che dalla memoria devono essere inviate al record di un disco e gli altri 256 sono riservati alle informazioni che dal disco devono essere inviate alla memoria.

4.3 – Tipi di informazioni registrabili

Le informazioni che possono essere registrate nella memoria ausiliaria di massa possono essere di due tipi:

- **programmi;**
- **dati numerici o stringa.**

I programmi, costituiti da un insieme di istruzioni simboliche (scritti in diversi linguaggi di programmazione), sono organizzati in modo sequenziale sia che si tratti di memoria a nastro sia che si tratti di memoria a disco.

Normalmente, questi file sono chiamati anche PROGRAM-FILE e sono individuabili per mezzo di un nome mnemonico, formato da un certo numero di caratteri alfanumerici.

Nel linguaggio BASIC, per registrare o leggere un program-file memorizzato su nastro o disco, si eseguono i due seguenti comandi BASIC, piú volte incontrati nelle precedenti lezioni:

SAVE "PROGRAMMA N. 1" per registrare
LOAD "PROGRAMMA N. 1" per leggere

dove "PROGRAMMA N. 1" è il nome mnemonico assegnato al programma BASIC da registrare.

I dati numerici o stringa sono organizzati in modo sequenziale se si tratta di memoria a nastro, in modo sequenziale o ad accesso diretto se si tratta di memoria a disco.

Normalmente, queste informazioni sono chiamate DATA-FILE, e sono individuabili anch'esse tramite nomi mnemonici.

La suddivisione dei data-file in record è valida, per i personal computer, soltanto per i file ad accesso diretto o random; ciò vuol dire che per i data-file sequenziali non esiste il record, bensí il campo, che può essere costituito da un valore numerico o da un valore stringa.

Nei file sequenziali i campi sono disposti uno di seguito all'altro e separati da speciali caratteri separatori come la virgola, il carattere di ritorno a capo **CR** o **RETURN** o ↵ (questo carattere ha codice ASCII decimale 13 e binario ØØØØ11Ø1).

Invece, nei data-file ad accesso diretto, dove esiste la suddivisione in record, questi ultimi possono essere a loro volta ripartiti in campi disposti uno di seguito all'altro e separati da speciali caratteri separatori.

Nei personal computer la lunghezza dei record espressa in byte è generalmente fissa, a differenza dei grandi sistemi di elaborazione dove si possono avere record a lunghezza fissa e variabile sia per le memorie a nastro magnetico sia per le memorie a disco.

Nella *fig. 10* sono rappresentate le strutture organizzative di data-file sequenziali e di data-file ad accesso diretto o random.

Nel data-file sequenziale (*fig. 10-a*) sono memorizzate le parole di una celebre poesia del Carducci: "Davanti a San Guido"; nel data-file random (*fig. 10-b*), sono contenute le informazioni relative ad un breve elenco telefonico personale.

Come si può notare, sotto ogni carattere registrato sono riportati i corrispondenti codici ASCII decimali (*fig. 9* della lezione *Analisi e Programmazione 2*) e le suddivisioni dei file.

Naturalmente i caratteri registrati nella memoria a disco non sono i codici decimali, bensí quelli binari (bit).

Nella struttura sequenziale i vari campi sono costituiti da singole parole, che sono disposte una di seguito all'altra e separate dal carattere ↵ , il cui codice ASCII decimale è 13.

Nella struttura random o ad accesso diretto i gruppi di campi sono registrati in singoli record, costituiti da un numero fisso di caratteri che, nel nostro esempio, viene assunto uguale a 100 byte.

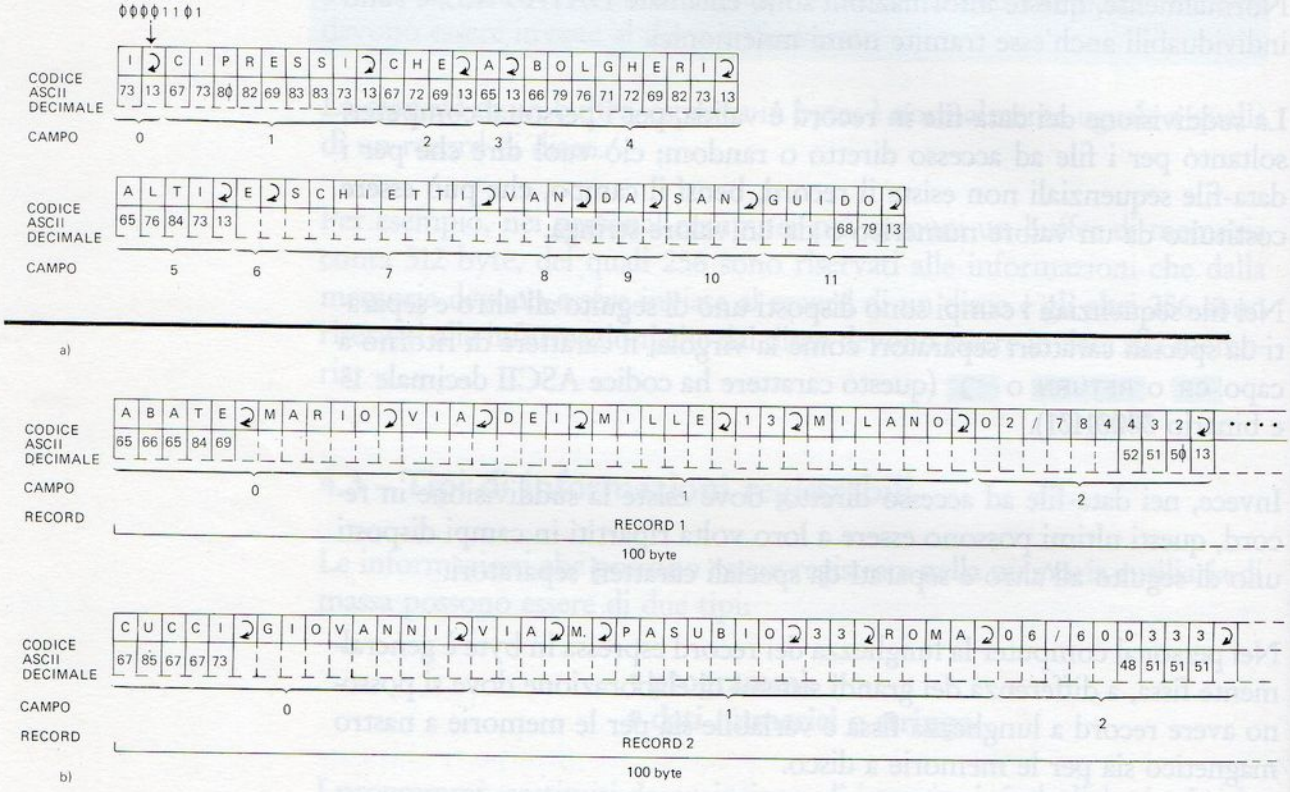


Fig. 10 – Organizzazione dei data-file sequenziali (a) e dei data-file random (b).

In tal modo, il metodo di memorizzazione a struttura random può risultare meno efficiente del metodo a struttura sequenziale, per quanto riguarda l'occupazione di memoria a disco (come si vede nella *fig. 10*, ogni record contiene spazi di caratteri inutilizzati), mentre risulta molto piú efficiente del primo per quanto riguarda l'accessibilità ad una singola parola o gruppo di parole. Ogni record del file ad accesso diretto è assimilabile ad un file ad accesso sequenziale.

Quando da programma si vuole ricercare un singolo campo contenuto in un certo record di un file, viene prima effettuata una ricerca veloce di quest'ultimo (record che contiene il campo) e poi si ricerca il campo specifico nell'ambito del record trovato (quest'ultima ricerca viene effettuata nella memoria principale del computer).

5. – Cenni sull'organizzazione delle informazioni nei grandi sistemi di elaborazione

Le memorie ausiliarie di massa, nei medio-grandi calcolatori elettronici, sono costituite da apparecchiature in grado di gestire una mole di supporti magnetici notevolmente superiore a quella dei piccoli sistemi di elaborazione.

Le bobine dei nastri magnetici sono piú voluminose ed i dischi sono costituiti da materiale plastico rigido, a differenza dei floppy-disk.

Nella *fig. 11* sono appunto visibili le unità a nastro magnetico ed a disco, utilizzate nei grandi centri di elaborazione dati.

Ogni bobina o disco può avere la capacità di milioni e milioni di caratteri; inoltre, ad uno stesso sistema di elaborazione possono essere collegate decine e decine di unità a nastro o a disco, rendendo la capacità di memoria ausiliaria praticamente illimitata.

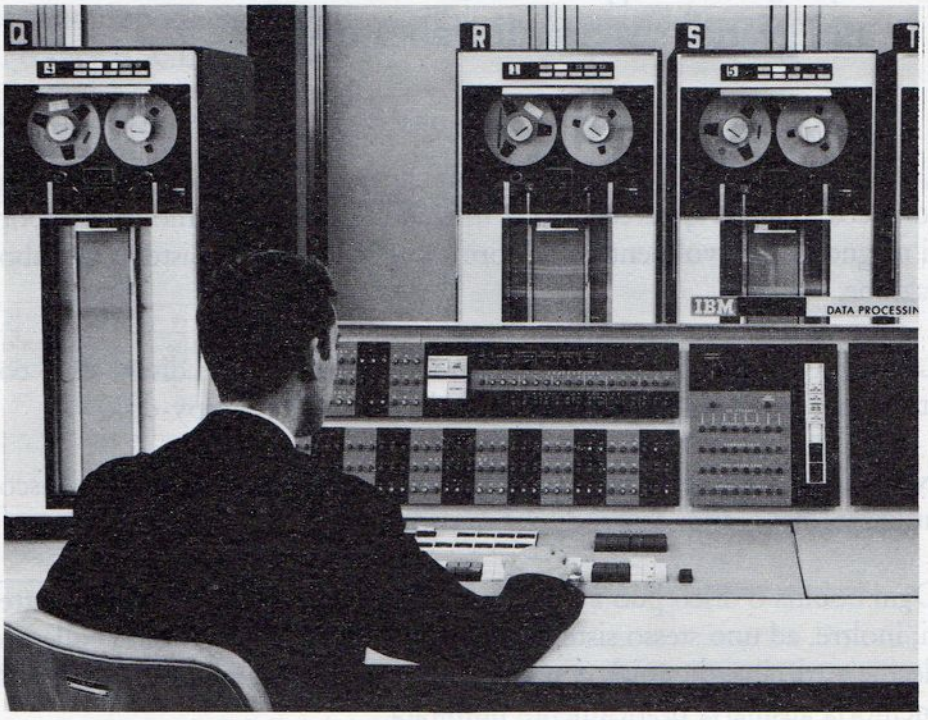
E' naturale, quindi, che in un sistema di calcolo del genere l'enorme mole di informazioni registrate sulla memoria di massa sia dotata di un'organizzazione adeguata, che non sia la semplice organizzazione sequenziale od anche la stessa sola organizzazione ad accesso diretto.

Nell'ultimo decennio sono stati sviluppati complessi sistemi di organizzazione delle informazioni (data-base); tramite questi sistemi è possibile, per esempio, elaborare e gestire le informazioni anagrafiche di migliaia di cittadini, i conti correnti bancari di tanti risparmiatori, compilare i libri paga di migliaia di dipendenti, ecc.

In un'azienda di certe dimensioni, tutte le informazioni riguardanti i prodotti da fabbricare ed i relativi componenti sono organizzate in strutture data-base del tipo presentato nella *fig. 4*, cioè "ad albero". Con il supporto di una o due unità a disco è possibile elaborare, gestire, modificare i dati di un certo numero di prodotti, in un tempo relativamente breve.

L'organizzazione dei dati su disco, sempre per i grandi sistemi di elaborazione, può essere anche **sequenziale con indice** (dall'inglese "index sequential"): con questa organizzazione ogni record o gruppo di informazioni è preceduto da una "chiave", che funge da indice di ricerca.

a)



b)

Fig. 11 – Unità a nastro (a) e a disco (b) impiegate nei grandi centri di elaborazione dei dati.

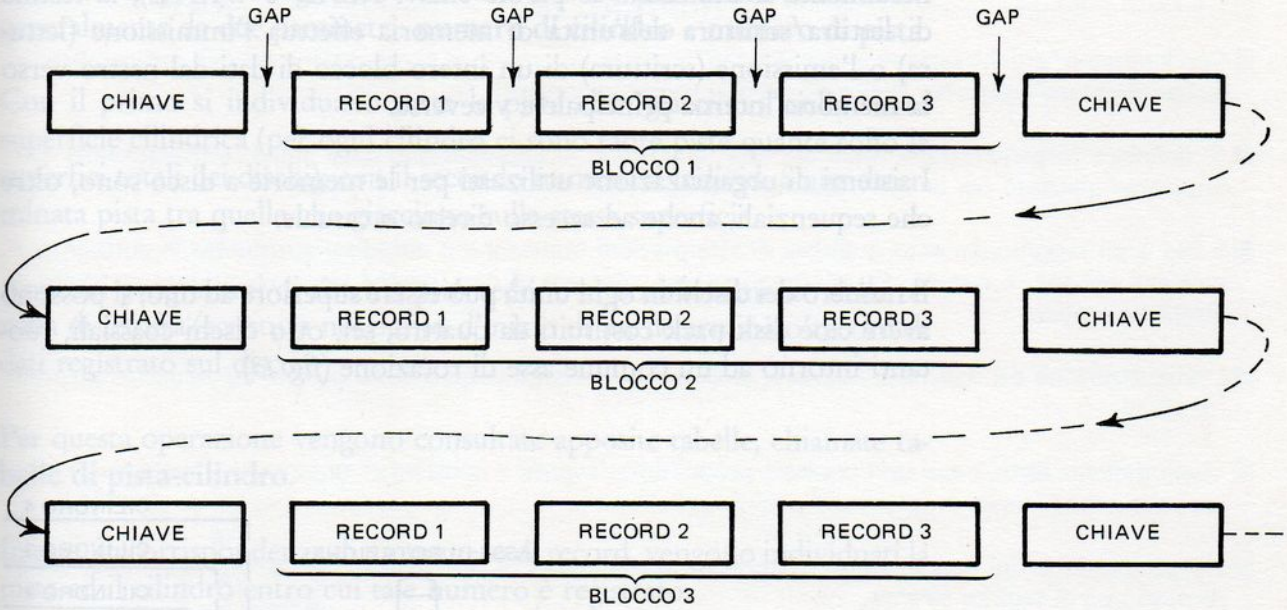


Fig. 12 – File di dati "index sequential", costituito da blocchi di tre record ciascuno, preceduti da codici o "chiavi" di ricerca.

La chiave è pure un'informazione e serve preliminarmente a verificare se quello che è registrato successivamente è proprio l'informazione specifica da elaborare oppure no.

Le chiavi di un file o archivio di dati sono in genere ordinate in modo crescente; in tal modo la ricerca delle singole informazioni avviene più velocemente.

La chiave come mezzo di ricerca delle informazioni

Il file di dati riportato nella *fig. 12* è un esempio di organizzazione sequenziale con indice, dove il campo CHIAVE funge da indice di ricerca delle informazioni.

In un file di dati sequenziali (con chiave o senza), i record possono essere costituiti da un numero fisso o variabile di caratteri (o byte).

Il file della *fig. 12* è costituito da record a lunghezza fissa, raggruppati in gruppi di tre ed ogni gruppo è chiamato **blocco**.

Nei grossi sistemi di calcolo, l'unità minima di lettura/scrittura è il blocco (gruppo di record) per cui, quando dall'interno di un programma viene eseguita un'istruzione di lettura/scrittura (per queste operazioni ge-

neralmente si utilizzano le parole chiavi READ e WRITE), la testina di lettura/scrittura dell'unità di memoria effettua l'immissione (lettura) o l'emissione (scrittura) di un intero blocco di dati dal nastro verso la memoria interna principale e viceversa.

I sistemi di organizzazione utilizzati per le memorie a disco sono, oltre che sequenziali, anche ad accesso diretto o casuale.

Il numero dei dischi in ogni unità può essere superiore ad uno; si possono avere cioè disk-pack costituiti da quattro, sei, otto dischi coassiali, ruotanti intorno ad un comune asse di rotazione (*fig. 13*).

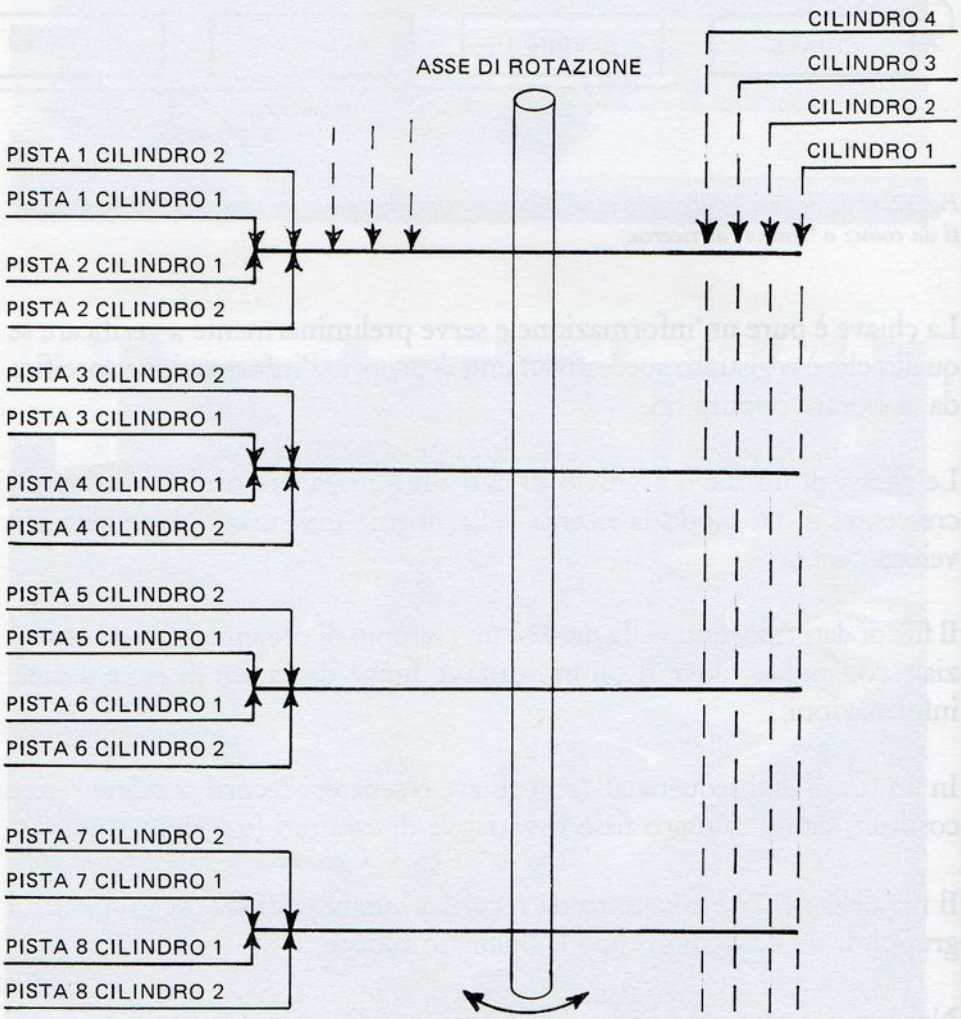


Fig. 13 – Rappresentazione schematica di un'unità a disco costituita da quattro dischi solidali.

Un record specifico registrato sull'unità a disco può essere individuato generalmente da due parametri: **numero di cilindro** e **numero di pista**.

Con il primo si individuano tutte le piste che "giacciono" sulla stessa superficie cilindrica (per ogni cilindro ci sono tante piste quante sono le superfici totali dei dischi); con il secondo numero si individua una determinata pista tra quelle che giacciono sulla stessa superficie.

Appositi programmi di sistema provvedono al posizionamento della testina di lettura/scrittura nel punto d'inizio di un determinato record di dati registrato sul disco.

Per questa operazione vengono consultate apposite tabelle, chiamate **tabelle di pista-cilindro**.

In esse, in corrispondenza di un numero di record, vengono individuati la pista ed il cilindro entro cui tale numero è registrato.

Dopo aver individuato pista e cilindro, il sistema attiva l'unità fisica di lettura/scrittura (testina elettromagnetica) per un suo rapido posizionamento sul record specifico da ricercare nell'ambito della pista.

* * * * * * * *

In seguito...

Nella prossima lezione verranno presi in considerazione, fra l'altro, i concetti fondamentali del calcolo numerico e statistico e numerose applicazioni della matematica al computer: sistemi di numerazione, algebra booleana, calcolo matriciale, funzioni, ecc.

Da memorizzare

- Con il termine "organizzazione logica dei dati" si indica un criterio di aggregazione dei dati **indipendente dal particolare supporto fisico** su cui essi sono memorizzati.
- Si definisce "organizzazione fisica dei dati" un criterio di aggregazione degli stessi, **condizionato dal supporto** utilizzato per la loro memorizzazione.
- Il **file** è un raggruppamento qualsiasi di informazioni numeriche e stringhe, strutturato in sottogruppi chiamati **record** e **campi**. E' un'organizzazione concreta di dati, perché viene supportato da nastri e dischi magnetici.
- L'organizzazione dei dati nella memoria ausiliaria può essere **sequenziale** oppure **ad accesso diretto** (random).
- Viene definito **data-base** uno speciale programma, avente il compito di organizzare, archiviare i dati secondo schemi logici programmati dall'utente.
- L'organizzazione dei dati su nastro magnetico è solamente di **tipo sequenziale**, cioè tutti i dati sono disposti uno di seguito all'altro.
- La velocità di trasferimento dei dati di un sistema, ossia la velocità con la quale essi possono venire letti o scritti nella memoria principale, è espressa in **byte al secondo**.
- L'organizzazione dei dati su disco può essere **sequenziale e ad accesso diretto**.
- Con il termine D.O.S. (Disk Operating System) viene definita la serie di procedure per la registrazione dei dati su disco.
- I floppy-disk possono essere del tipo **hard-sectored**, i cui settori sono contrassegnati con picccoli fori, e del tipo **soft-sectored**, con settori identificati mediante apposite informazioni registrate sulla traccia.
- Il **buffer** è una zona temporanea di stazionamento della memoria principale, attraverso cui transitano le informazioni che dalla memoria devono essere inviate al disco e viceversa.
- Nell'organizzazione dei dati nel sistema sequenziale con indice ogni record, o gruppo di informazioni, è preceduto da una "chiave", che funge da indice di ricerca.

Scuola Radio Elettra
Torino

Pratica del BASIC

Lezione 8

1 — La gestione di un piccolo archivio di informazioni
2 — Ordinamento di una serie di stringhe
3 — Effetti di animazione sul video
4 — Grafica ed animazione con **POKE** e **PEEK** sul C64
5 — Grafica ed animazione con **POKE** e **PEEK** sul VIC-20

Che cosa farà

Questa lezione è praticamente costituita da due sezioni o parti.

Nella prima parte verranno esaminati due interessanti programmi, relativi rispettivamente alla gestione di un piccolo archivio di informazioni (mediante l'uso delle istruzioni **READ** e **DATA**) e all'ordinamento alfabetico di una serie di stringhe (nominativi).

Nella seconda parte della lezione si analizzeranno i concetti basilari per utilizzare le capacità grafiche del C64 e del VIC-20, al fine di visualizzare sullo schermo figure in movimento (effetti di animazione) mediante l'uso dei comandi del cursore e tramite le istruzioni **POKE** e **PEEK**. Queste istruzioni, consentendo di accedere direttamente alla memoria del computer, sono frequentemente usate nella programmazione dei video-giochi.

1. - La gestione di un piccolo archivio di informazioni

Come già precedentemente precisato, l'istruzione **READ** serve a richiamare dati dall'"interno" di un programma, dati che, raggruppati ed elencati nelle apposite istruzioni **DATA**, costituiscono un vero e proprio archivio interno.

Questo archivio di informazioni (numeriche o stringa) è organizzato, all'interno del programma, in modo sequenziale, tant'è che esso è assimilabile ad un file di nastro magnetico o ad un vettore di memoria principale.

Naturalmente, l'accessibilità e la reperibilità delle informazioni all'interno del programma sono piú immediate che non sul supporto di memoria magnetica.

L'istruzione **DATA** contiene l'elenco sequenziale delle informazioni e funge cosí da deposito o archivio interno di informazioni, che possono essere sia valori numerici (non racchiusi tra virgolette) sia valori stringa (racchiusi tra virgolette se al loro interno contengono uno o piú spazi, virgole e due punti).

Ciascuna informazione contenuta nell'istruzione **DATA** è separata da quella che segue da una virgola, che funge quindi da carattere divisorio.

I valori elencati nell'istruzione **DATA** sono immessi nella memoria principale attraverso l'istruzione **READ**, seguita da un elenco di nomi di variabili (nomi corrispondenti ai singoli valori che vanno immessi).

Se, per esempio, l'elenco **DATA** è cosí composto:

```
500 DATA 3.14159, "ROSSI CARLO", SCUOLA
```

e si vogliono immettere le tre suddette informazioni nella memoria principale, occorre scrivere la **READ** nel seguente modo:

```
10 READ P, FB$, SC$
```

cioè assegnare il nome P al valore numerico 3.14159 (prima informazione dell'elenco DATA), il nome FB$ al nominativo "ROSSI CARLO" che, contenendo uno spazio all'interno, va racchiuso tra virgolette, ed il nome SC$ alla parola SCUOLA, da cui possono essere omesse le virgolette.

Come si può vedere, l'istruzione **DATA** prima scritta ha un numero di riferimento piuttosto alto. Infatti, è buona regola riservare le ultime posizioni di un programma alle istruzioni archivio DATA, se non altro perché in tal modo possono essere prontamente evidenziate ed eventualmente modificate.

Trattandosi di un archivio, è possibile cioè aggiungere nuovi elenchi DATA con numeri di riferimento sempre crescenti.

Un esempio di archivio interno di dati è rappresentato nella *fig. 1*, dove è codificato in BASIC un semplice programma che preleva informazioni da elenchi DATA e li visualizza sullo schermo.

In tali elenchi DATA sono riportate informazioni relative a persone di cui si vuole ricordare:

1) cognome e nome;
2) via e numero civico;
3) C.A.P. e città;
4) numero telefonico.

E' sempre possibile aggiungere altri nominativi all'elenco già esistente: basta scrivere altri elenchi DATA con numeri di riferimento sempre crescenti. Naturalmente, occorre tenere presente che la memoria disponibile del computer non è inesauribile!

Nel C64, per esempio, sono disponibili 38911 Kbyte per memorizzare dati e programmi, mentre nel computer VIC-20 il loro numero è sensibilmente inferiore: sono disponibili esattamente 3583 Kbyte.

Come si vede sulla *fig. 1*, l'ultimo elenco DATA contiene il valore numerico "999" rappresentato come stringa: questo valore sta ad indicare la fine dell'archivio di informazioni e viene utilizzato dal programma per inter-

rompere l'elaborazione.

Per semplificare l'esempio, il programma della *fig. 1* contiene un solo nominativo (ABATE PAOLO) riportato fra le istruzioni 15Ø e 16Ø.

Naturalmente è possibile aggiungere altri nominativi, partendo dal numero di linea 164 e spostando la corrispondente DATA in fondo al nuovo elenco, come ultima istruzione.

```
10 REM******************
12 REM* INDIRIZZARIO *
15 REM******************
20 PRINT"□";
25 READ I$,D$
27 PRINT"  ********************"
30 PRINT "  *";I$;D$;"*"
35 PRINT"  ********************"
40 R=1
45 FOR IR=1 TO 3
50 READ N$,I$,C$,T$
55 IF N$="999" THEN 105
60 PRINT N$
63 PRINT I$
65 PRINT C$
67 PRINT T$
68 PRINT"------------------------"
70 NEXT IR
75 PRINT"BATTI UN TASTO QUALSIASI PER CONTINUARE";
80 GET A$
85 IF A$="" THEN 80
87 PRINT"□"
90 GOTO 40
105 PRINT"FINE ARCHIVIO CLIENTI"
108 PRINT I$
110 END
140 DATA"CLIENTI"," 01/01/85"
150 DATA"ABATE PAOLO"
154 DATA"VIA MONTE PASUBIO 133"
156 DATA"10124 TORINO"
160 DATA"TEL.011/778899"
164 DATA"999","AGGIUNGERE DA 164","*","*"
```

Fig. 1 – Listato del programma che "gestisce" un indirizzario di persone (clienti, amici, fornitori, ecc.).

In altre parole, al termine dell'elenco si dovrà riportare l'istruzione:

X **DATA** "999", "AGGIUNGERE DA X","*","*"

dove con X viene indicato il numero dell'ultima istruzione **DATA** scritta nel programma.

Inoltre, è possibile utilizzare altri nomi e date nella intestazione dell'archivio, modificando l'istruzione 14Ø.

Se Lei, per esempio, vuole conservare indirizzi di amici e datare dall'inizio di un certo anno l'archivio, può scrivere l'istruzione 14Ø in questo modo:

14Ø **DATA** "AMICI"," Ø1/Ø1/85"
↑
spazio

In tal caso abbia cura di inserire uno spazio tra le virgolette di apertura della data dell'archivio e l'inizio della data stessa.

Vediamo ora le fasi operative piú importanti del programma.

Nell'istruzione 25 le variabili di nome I$ e D$ vengono associate alle due corrispondenti stringhe, "CLIENTI" e "Ø1/Ø1/85" dell'istruzione **DATA** riportata nella linea di programma 14Ø, le quali vengono poi visualizzate sullo schermo con le istruzioni **PRINT** 27÷35.

Il ciclo **FOR... NEXT...** (istruzioni 45÷7Ø) consente di visualizzare in un'unica videata le informazioni relative al massimo a tre persone.

Le istruzioni 6Ø÷67 provvedono a stampare le stringhe N$ (nome e cognome), I$ (via e numero civico), C$ (C.A.P. e nome della città) e T$ (numero telefonico). Tali stringhe sono contenute nelle corrispondenti istruzioni **DATA** 15Ø, 154, 156, 16Ø.

L'istruzione 68 provvede ad interporre fra ciascuna informazione che appare sul video e la successiva una serie di trattini orizzontali.

Utilizzando il computer VIC-20, tali trattini dovranno essere 22; se si impiega invece il computer C64, essi dovranno essere 40.

Al termine di ogni videata, per passare alla successiva è sufficiente preme-

re un tasto qualsiasi della tastiera.

Ciò viene ottenuto mediante l'istruzione **GET** (linea di programma 8Ø), che consente di immettere un carattere (A$) per volta senza premere il tasto **RETURN**.

Se nessun tasto viene premuto (istruzione 85), ossia se A$ vale "", il programma ritorna alla linea 8Ø e rimane in attesa sino a che non viene premuto un tasto qualsiasi.

Quando la variabile N$="999", cioè quando si è giunti al termine della lista degli indirizzi, viene stampato il messaggio "FINE ARCHIVIO CLIENTI".

Come si può rilevare, questo programma è molto semplice, poiché consente di leggere e stampare le informazioni relative a persone senza effettuare elaborazioni.

Terminato l'esame del listato, batta sulla tastiera il programma e lo esegua con il comando **RUN**.

Dopo averlo verificato, il programma può essere registrato su una cassetta nastro con il nome stesso dell'archivio ("CLIENTI", "AMICI", "FORNITORI", "AMICHE", ecc.), seguendo sempre la stessa procedura operativa, già menzionata piú volte, per l'esatto posizionamento del nastro dopo l'ultimo file-programma registrato.

Riporti inoltre le relative note informative sul foglio-archivio nastro descritto nella *Pratica del BASIC 2*.

Per effettuare un rapido posizionamento del nastro su un determinato file registrato, si può ricorrere ad un metodo pratico e veloce.

Come avrà potuto constatare, sull'apparecchio registratore c'è un contatore con tre cifre ed un tasto azzeratore (*fig. 2*). Il contatore conta i giri

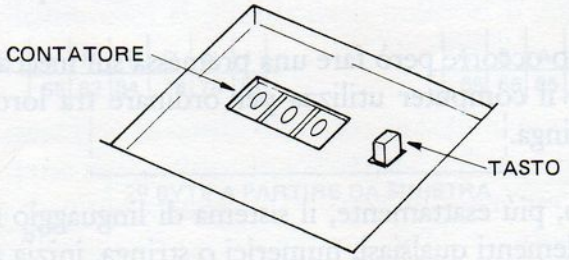


Fig. 2 – Contatore del registratore.

effettuati dalla bobina del nastro durante l'operazione di lettura o scrittura (registrazione).

E' possibile quindi conoscere la posizione di ogni file-programma sul nastro, se si ha l'accortezza di segnare sul foglio-archivio il numero che si legge sul contatore, **numero che va letto però prima di effettuare la registrazione del file sul nastro.**

Conoscendo il numero del contatore corrispondente ad un certo file-programma, si può effettuare un posizionamento manuale veloce (azzerando preventivamente il contatore e premendo il tasto di avanzamento veloce F.F WD del registratore) sui file-programmi che precedono immediatamente quello da ricercare.

Per ottenere ciò è sufficiente bloccare l'avanzamento (con il tasto STOP) qualche numero prima di quello corrispondente al file da ricercare.

Dopo di ciò si può effettuare il comando BASIC

VERIFY "NOME FILE"

in modo da posizionare la testina di lettura/scrittura esattamente dopo il file-programma che ha il nome riportato a destra del comando **VERIFY**.

2. - Ordinamento di una serie di stringhe

Nella precedente lezione avevamo elaborato un programma BASIC che eseguiva l'ordinamento crescente o decrescente di una serie qualsiasi di valori numerici.

In questa lezione proveremo ad ordinare una serie di parole o stringhe.

A questo punto occorre però fare una premessa sui meccanismi logici di confronto che il computer utilizza per ordinare tra loro piú elementi numerici o stringa.

Il computer (o, piú esattamente, il sistema di linguaggio BASIC), nel raffrontare due elementi qualsiasi, numerici o stringa, inizia a valutare i valori dei codici ASCII dei byte posti piú a sinistra.

Se da questo iniziale confronto risulta che il codice del primo byte del primo elemento è inferiore o superiore al codice del primo byte del secondo elemento, il computer stabilisce che il primo elemento è inferiore o superiore al secondo elemento; se invece il primo byte del primo elemento risulta uguale al primo byte del secondo elemento, il confronto prosegue sui secondi byte dei rispettivi elementi.

Il procedimento di confronto (per byte) continua fino a quando non sono stati esauriti tutti i caratteri degli elementi.

Per esempio, nel caso dei due numeri riportati nella *fig. 3-a*, il primo risulta inferiore al secondo al quarto confronto dei rispettivi byte; nel caso invece delle due stringhe della *fig. 3-b*, la prima risulta maggiore (o successiva) della seconda al secondo confronto dei rispettivi byte.

Nel caso numerico si tiene conto, naturalmente, del segno che precede il numero e del suo ordine di grandezza. L'esempio riportato nella *fig. 3-a* si riferisce a due numeri interi aventi lo stesso numero di cifre, cioè lo stesso ordine di grandezza.

In pratica, il procedimento esposto è analogo a quello che noi tutti seguiamo quando dobbiamo ricercare un nominativo sull'elenco telefonico: il nominativo, memorizzato nella nostra memoria, viene raffrontato carattere per carattere con quelli riportati sull'elenco ed il confronto inizia dai caratteri posti piú a sinistra procedendo verso quelli posti piú a destra.

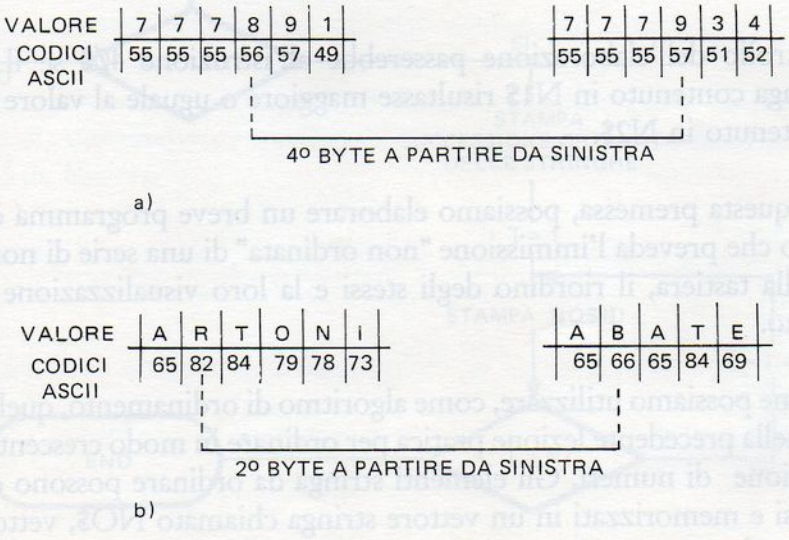


Fig. 3 – Confronti fra elementi numerici (a) ed elementi stringa (b).

L'istruzione BASIC che permette di indagare se un elemento è inferiore (cioè precede) o se è superiore (cioè segue) ad un altro è la seguente:

IF A (REL) B **THEN** num. linea

dove (REL) sta ad indicare uno dei seguenti operatori relazionali:

=	uguale
>	maggiore
<	minore
>=	maggiore o uguale
<=	minore o uguale
<>	diverso

e "num. linea" un numero di riferimento di istruzione BASIC a cui viene passato il controllo dell'elaborazione se (IF) la relazione di confronto tra A e B risulta "VERA"; A e B possono essere o nomi di variabili o costanti.

Ad esempio, se scrivessimo in un programma:

```
200 IF A<B THEN 300
```

il controllo dell'elaborazione passerebbe all'istruzione 3ØØ se il valore numerico di A risultasse minore del valore numerico di B.

Se scrivessimo ancora:

```
200 IF N1$>=N2$ THEN 400
```

il controllo dell'elaborazione passerebbe all'istruzione 4ØØ se il valore stringa contenuto in N1$ risultasse maggiore o uguale al valore stringa contenuto in N2$.

Dopo questa premessa, possiamo elaborare un breve programma dimostrativo che preveda l'immissione "non ordinata" di una serie di nominativi dalla tastiera, il riordino degli stessi e la loro visualizzazione sullo schermo.

A tal fine possiamo utilizzare, come algoritmo di ordinamento, quello già usato nella precedente lezione pratica per ordinare in modo crescente una successione di numeri. Gli elementi stringa da ordinare possono essere immessi e memorizzati in un vettore stringa chiamato NO$, vettore di stringhe che va appunto ordinato in modo crescente.

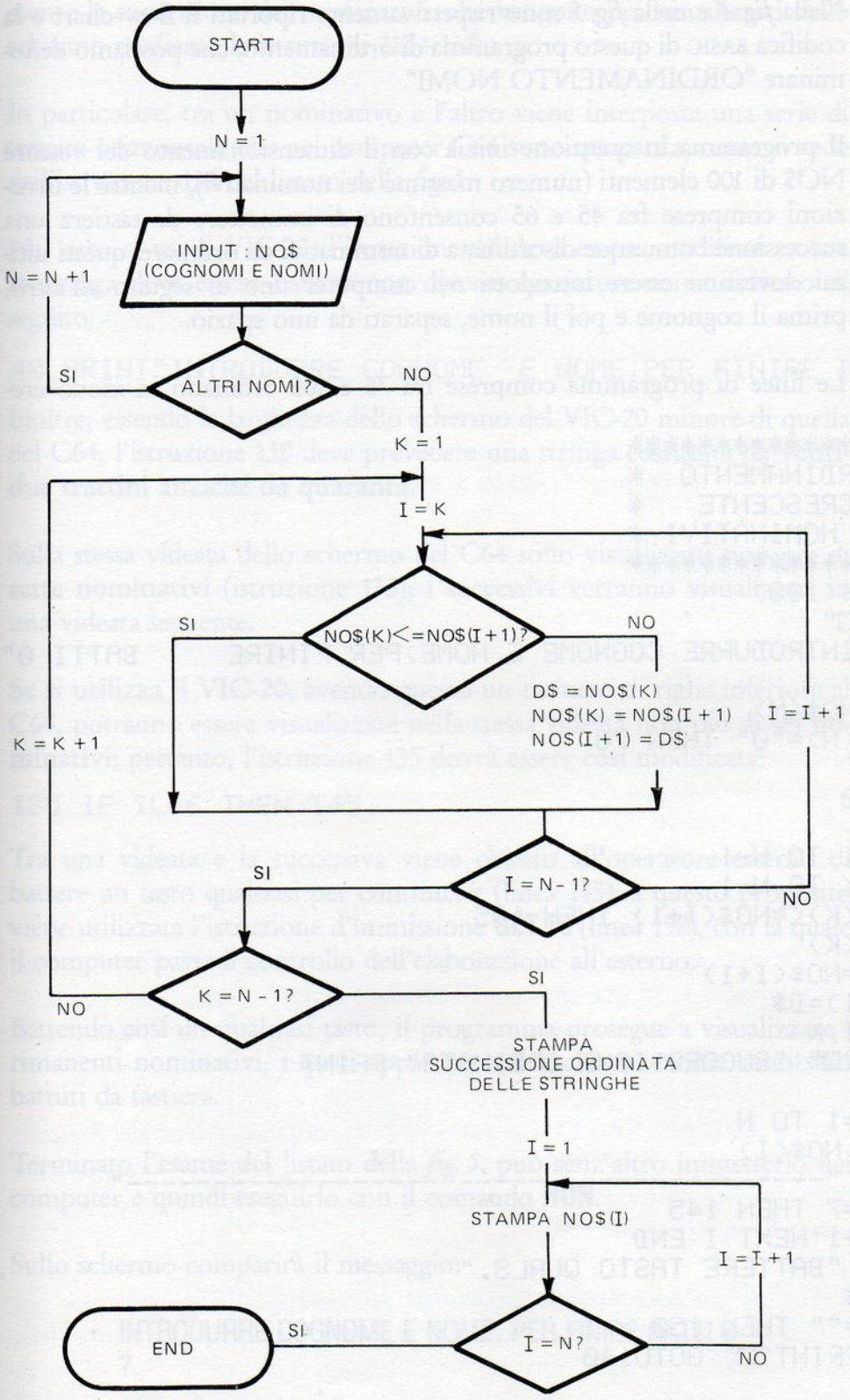


Fig. 4 – Flow-chart del programma "ORDINAMENTO NOMI".

Nella *fig. 4* e nella *fig. 5* sono rispettivamente riportati il flow-chart e la codifica BASIC di questo programma di ordinamento, che possiamo denominare "ORDINAMENTO NOMI".

Il programma in questione inizia con il dimensionamento del vettore NO$ di 100 elementi (numero massimo dei nominativi), mentre le istruzioni comprese fra 45 e 65 consentono di immettere da tastiera una successione comunque disordinata di nominativi da ordinare: questi ultimi dovranno essere introdotti nel computer uno di seguito all'altro, prima il cognome e poi il nome, separati da uno spazio.

Le linee di programma comprese fra 75 e 105 ordinano in modo cre-

```
10 REM*****************
15 REM* ORDINAMENTO  *
20 REM*  CRESCENTE   *
25 REM*DI NOMINATIVI *
30 REM*****************
32 DIM NO$(100)
35 PRINT"□"
40 PRINT"INTRODURRE COGNOME E NOME;PER FINIRE      BATTI 0"
45 N=1
50 INPUT NO$(N)
55 IF NO$(N)="0" THEN 70
60 N=N+1
65 GOTO 50
70 N=N-1
75 FOR K=1 TO N-1
80 FOR I=K TO N-1
85 IF NO$(K)<=NO$(I+1) THEN 105
90 D$=NO$(K)
93 NO$(K)=NO$(I+1)
95 NO$(I+1)=D$
105 NEXT I,K
110 PRINT"□";"SUCCESSIONE ORDINATA":PRINT
115 IL=1
120 FOR I=1 TO N
125 PRINT NO$(I)
130 PRINT "-----------------------------------------"
135 IF IL=7 THEN 145
140 IL=IL+1:NEXT I:END
145 PRINT "BATTERE TASTO QUALS."
150 GET T$
155 IF T$="" THEN 150
160 IL=0:PRINT"□":GOTO140
165 END
```

Fig. 5 – Codifica del programma di ordinamento di una successione di nominativi.

scente la successione dei nominativi, che vengono poi visualizzati sullo schermo mediante le istruzioni 11Ø ÷ 13Ø.

In particolare, tra un nominativo e l'altro viene interposta una serie di trattini orizzontali, che nel computer C64 **sono quaranta**, come si vede dall'istruzione 13Ø del listato della *fig. 5*.

Nel caso si usi il VIC-20, l'istruzione 4Ø deve essere modificata, ossia la stringa deve essere battuta con diverse spaziature come indicato di seguito:

```
40 PRINT"INTRODURRE COGNOME  E NOME;PER FINIRE BATTI Ø"
```

Inoltre, essendo la larghezza dello schermo del VIC-20 minore di quella del C64, l'istruzione 13Ø deve prevedere una stringa costituita **da ventidue trattini anziché da quaranta.**

Sulla stessa videata dello schermo del C64 sono visualizzati non piú di **sette nominativi** (istruzione 135); i successivi verranno visualizzati in una videata seguente.

Se si utilizza il VIC-20, avendo questo un numero di righe inferiore al C64, potranno essere visualizzate nella stessa videata non piú di **sei nominativi**; pertanto, l'istruzione 135 dovrà essere cosí modificata:

```
135 IF IL=6 THEN 145
```

Tra una videata e la successiva viene chiesto all'operatore esterno di battere un tasto qualsiasi per continuare (linea 145); a questo proposito viene utilizzata l'istruzione d'immissione **GET** T$ (linea 15Ø), con la quale il computer passa il controllo dell'elaborazione all'esterno.

Battendo cosí un qualsiasi tasto, il programma prosegue a visualizzare i rimanenti nominativi, i quali appaiono sullo schermo come sono stati battuti da tastiera.

Terminato l'esame del listato della *fig. 5*, può senz'altro immetterlo nel computer e quindi eseguirlo con il comando **RUN**.

Sullo schermo comparirà il messaggio:

```
INTRODURRE COGNOME E NOME; PER FINIRE BATTI Ø
?
```

Il punto interrogativo segnala che devono essere introdotti il cognome

ed il nome, separati fra loro da uno spazio.

Operando con il VIC-20, è opportuno abbreviare i nominativi la cui lunghezza complessiva (cognome+nome) sia superiore a venti caratteri, per evitare riporti a capo.

Ad ogni nominativo battuto comparirà sul video un punto interrogativo.

Per terminare l'immissione dei nominativi occorre battere il tasto Ø, seguito naturalmente dal tasto **RETURN**. Il computer visualizzerà in ordine alfabetico i nominativi immessi.

Potrà notare che, immettendo alcune decine di nominativi, il computer impiega diversi secondi per eseguire l'ordinamento previsto.

3. – Effetti di animazione sul video

Nella precedente lezione ha avuto modo di constatare praticamente come viene ottenuta la simulazione di un effetto di movimento (detto anche **animazione**), realizzando un programma con il quale si produceva l'immagine di un aereo che si spostava sullo schermo.

Ora approfondiremo le conoscenze nella tecnica di animazione, che trova interessanti e vasti impieghi nei video-giochi.

Per provocare l'illusione del movimento delle immagini sullo schermo del televisore si adotta una tecnica simile a quella utilizzata per i cartoni animati, la quale, come è noto, consiste nel sostituire rapidamente una immagine con un'altra leggermente diversa dalla precedente.

I computer C64 e VIC-20 possiedono una vasta gamma di caratteri grafici (memorizzati nella ROM), con i quali si può realizzare una quantità praticamente illimitata di figure (grafica in bassa risoluzione) che possono essere facilmente animate.

Come già precedentemente constatato, i simboli grafici da utilizzare per la realizzazione delle immagini sono riportati sulla parte anteriore della maggior parte dei tasti.

Ricordi a questo proposito che, per visualizzare sullo schermo il simbolo

grafico riportato a sinistra del tasto, occorre tenere premuto il tasto e, mantenendolo abbassato, battere il tasto del segno grafico desiderato. Per visualizzare il segno grafico riportato a destra del tasto si deve invece premere il tasto **SHIFT** e, mantenendolo premuto, abbassare il tasto interessato.

Vediamo ora un interessante esempio di programma, che crea la figura molto semplice di un uomo le cui braccia e gambe sono in movimento, simulando cosí, con un pò di fantasia, l'immagine di un "atleta" in allenamento.

L'immagine viene creata utilizzando determinati caratteri grafici del computer, opportunamente sovrapposti fra loro.

Il listato del programma in questione è riportato nella *fig. 6*.

L'istruzione 15 provvede, come è noto, a cancellare lo schermo ed a posizionare il cursore in alto a sinistra.

Le linee di programma 2Ø÷4Ø realizzano, mediante istruzioni **PRINT**, la prima immagine dell'uomo; quindi, l'istruzione 5Ø riposiziona il cursore in alto a sinistra dello schermo prima di formare la seconda immagine

```
10 REM*  L'ATLETA  *
15 PRINT"▯"
20 PRINT" ▚O "
25 PRINT"  ▒▔"
30 PRINT"  ▒▒ "
35 PRINT" ▒▌▀ ▔"
40 PRINT" ▔  "
45 FOR T=1 TO 300:NEXT T
50 PRINT"▒"
55 PRINT"  O "
60 PRINT" ▔▒▚"
65 PRINT"  ▒▒ "
70 PRINT" ▔ ▌"
75 PRINT"   ▔"
80 FOR I=1 TO 300:NEXTI
85 GOTO 15
```

Fig. 6 – Programma che produce sullo schermo l'immagine di un uomo che muove le braccia e le gambe.

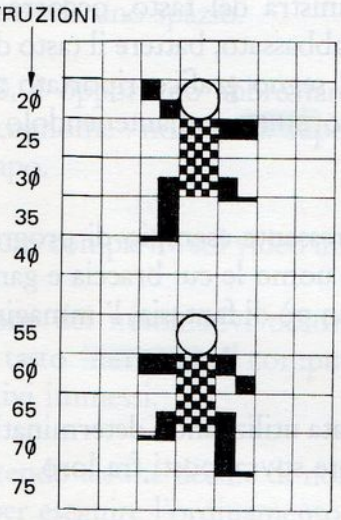


Fig. 7 – Raffigurazione dell'atleta in movimento.

dell'uomo stesso (leggermente diversa dalla prima per la posizione delle braccia e delle gambe) tramite le istruzioni **PRINT** 55÷75 (*fig. 7*).

In questo modo i caratteri grafici costituenti la seconda immagine vengono a sovrapporsi perfettamente ai precedenti caratteri cancellandoli, e dando cosí l'illusione che le braccia e le gambe dell'uomo si muovano.

Le istruzioni 45 e 8Ø creano i cicli di ritardo (pause di esecuzione), necessari per poter percepire chiaramente il movimento.

Infine, l'istruzione 85 fa ritornare l'esecuzione del programma all'inizio (linea 15).

Prima di procedere all'immissione del programma nel computer, vediamo, per ciascuna stringa di caratteri delle istruzioni **PRINT**, quali tasti grafici devono essere battuti per ottenere le due immagini previste: ricordi che gli spazi presenti nelle stringhe di caratteri, indicati con ␣, sono ottenuti con la barra spaziatrice.

PRIMA IMMAGINE DELL'UOMO

20 PRINT " 1␣ C= → B SHIFT → W 1␣"
25 PRINT " 2␣ C= → + C= → U "
30 PRINT " 1␣ C= → D C= → + C= → F "
35 PRINT " 1␣ CTRL → 9 C= → K CTRL → Ø 1␣ C= → T "
40 PRINT " 1␣ C= → T 2␣"

SECONDA IMMAGINE DELL'UOMO

```
55 PRINT " 2↓ SHIFT → W 1↓"
60 PRINT " 1↓ CTRL → U CTRL → + CTRL → B "
65 PRINT " 1↓ CTRL → D CTRL → + CTRL → F "
70 PRINT " 1↓ CTRL → T 1↓ CTRL → K "
75 PRINT " 3↓ CTRL → T "
```

Esaminati i diversi caratteri impiegati per la composizione delle due immagini, può senz'altro passare a battere sulla tastiera del computer il listato riportato nella *fig. 6*.

Mandando in esecuzione il programma con il comando **RUN**, sull'angolo superiore sinistro dello schermo apparirà l'immagine in movimento. Potrà variare a Suo piacere la velocità di movimento, cambiando il valore finale del ciclo di ritardo (3ØØ) delle istruzioni 45 e 8Ø; tenga presente che **aumentando tale valore, il movimento viene rallentato, mentre diminuendolo il movimento viene accelerato.**

Per fermare l'immagine dovrà premere il tasto **RUN | STOP**.

Vediamo ora un altro esempio di animazione, per il quale si utilizzano sempre i caratteri grafici predefiniti, che producono sullo schermo l'immagine di un satellite artificiale ruotante, il cui aspetto è rappresentato nella *fig. 8*.

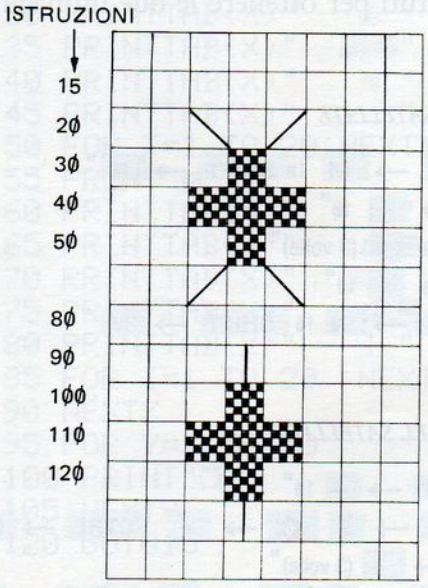


Fig. 8 – Raffigurazione del satellite artificiale ruotante.

```
10 REM*  SATELLITE  *
12 PRINT"■"
15 PRINT"   ▚ ▞"
20 PRINT"    ▒ "
30 PRINT"   ▒▒▒"
40 PRINT"    ▒ "
50 PRINT"   ▞ ▚"
60 FOR T=1 TO 20:NEXT T
70 PRINT"■"
80 PRINT"    | "
90 PRINT"    ■■ "
100 PRINT"   ▒▒▒"
110 PRINT"    ■■ "
120 PRINT"    |  "
125 FOR I=1 TO 20:NEXT I
130 GOTO 12
```

Fig. 9 – Programma che rappresenta sullo schermo la figura di un satellite artificiale ruotante.

Il listato del programma in questione è riportato nella *fig. 9*; come può osservare, esso non presenta particolarità degne di nota rispetto al precedente programma, ad eccezione naturalmente dei diversi caratteri grafici utilizzati nelle stringhe delle istruzioni **PRINT**, che adesso devono consentire la composizione delle due immagini del satellite artificiale.

Anche in questo caso, prima di procedere all'immissione del programma nel computer, è opportuno specificare chiaramente quali tasti grafici dovranno essere battuti per ottenere le due immagini da sovrapporre.

PRIMA IMMAGINE DEL SATELLITE

15 PRINT " 2♭ SHIFT → M 1♭ SHIFT → N "
20 PRINT " 3♭ C= → + 1♭"
30 PRINT " 2♭ C= → + (3 volte)"
40 PRINT " 3♭ C= → + 1♭"
50 PRINT " 2♭ SHIFT → N 1♭ SHIFT → M "

SECONDA IMMAGINE DEL SATELLITE

80 PRINT " 3♭ SHIFT → — 1♭"
90 PRINT " 3♭ CTRL → 9 C= → + CTRL → Ø 1♭"
100 PRINT " 2♭ C= → + (3 volte)"
110 PRINT " 3♭ CTRL → 9 C= → + CTRL → Ø 1♭"
120 PRINT " 3♭ SHIFT → — 2♭"

Dopo aver esaminato i caratteri grafici che devono essere utilizzati, cancelli dalla memoria del computer il precedente programma mediante il comando **NEW** e ripulisca lo schermo dalle scritte premendo contemporaneamente i tasti **RUN | STOP** e **RESTORE**.

Batta poi sulla tastiera del computer il listato riportato nella *fig. 9*, quindi esegua il programma con il comando **RUN**: sull'angolo superiore sinistro dello schermo apparirà l'immagine, naturalmente molto semplificata, di un satellite ruotante.

Vediamo ora come sia possibile effettuare lo spostamento dell'immagine del satellite da sinistra a destra sullo schermo.

Come ha già constatato in precedenza, per simulare lo spostamento di una immagine viene utilizzata la funzione **TAB** (X), la quale provvede a posizionare il cursore in corrispondenza della colonna dello schermo indicata dal valore numerico attribuito alla variabile X.

Nella *fig. 10* è rappresentato il listato del programma denominato "*Satellite in volo*", con cui si realizza appunto lo spostamento del satellite.

```
10 REM*  SATELLITE  *
12 REM*    IN VOLO   *
15 FOR X=0 TO 33
20 PRINT"□"
25 PRINTTAB(X)"  \ /"
30 PRINTTAB(X)" ▒ "
35 PRINTTAB(X)"▒▒▒▒▒"
40 PRINTTAB(X)" ▒ "
45 PRINTTAB(X)"  / \"
50 FOR T=1 TO 20:NEXTT
55 PRINT"■"
60 PRINTTAB(X)"  | "
65 PRINTTAB(X)" ▒▒▒ "
70 PRINTTAB(X)"▒▒▒▒▒"
75 PRINTTAB(X)" ▒▒▒ "
80 PRINTTAB(X)"  | "
85 FOR I=1 TO 20 :NEXTI
90 NEXTX
95 FOR Y=1 TO 50
100 PRINT"□"
105 NEXTY
120 GOTO15
```

Fig. 10 – Programma che raffigura sullo schermo un satellite ruotante che si sposta da sinistra a destra.

Nel programma in questione, l'istruzione 15 inizia un ciclo di elaborazione che viene eseguito sino a quando X (variabile della funzione **TAB**), partendo da Ø, raggiunge il numero 33: le istruzioni eseguite sono tutte quelle comprese fra la linea 2Ø e la linea 9Ø.

In questo modo il satellite ruotante viene fatto spostare di una posizione per volta dalla colonna Ø alla colonna 33.

Se si utilizza il computer VIC-20, poiché quest'ultimo ha un minore numero di colonne dello schermo, l'istruzione 15 dovrà essere così modificata:

```
15 FOR X=Ø TO 15
```

Il ciclo costituito dalle istruzioni 95 e 1Ø5 ha lo scopo di ottenere una breve pausa prima che il satellite, giunto sul lato destro dello schermo, ricompaia nuovamente sul lato sinistro per poi riprendere da capo lo spostamento.

Dopo avere riportato il computer nelle condizioni iniziali mediante il comando **NEW** ed i tasti RUN | STOP e RESTORE, batta il listato del programma riportato nella *fig. 10* e lo esegua con il comando **RUN**: vedrà il satellite ruotante spostarsi gradualmente da sinistra a destra dello schermo. Per fermare il programma prema il tasto RUN | STOP.

Dai semplici esempi realizzati ha avuto la possibilità di apprendere i principi fondamentali della tecnica di animazione. A questo punto, seguendo la traccia fornitaLe, potrà personalmente creare, secondo la Sua fantasia, figure animate utilizzando i caratteri grafici disponibili sul computer.

Naturalmente le immagini che vorrà produrre dovranno prima essere disegnate su un un foglio a quadretti (come mostrato nella *fig. 7* e nella *fig. 8*), facendo corrispondere ad ogni quadretto (posizione dello schermo) uno dei simboli grafici della tastiera.

Disegnata l'immagine desiderata, potrà utilizzare l'istruzione **PRINT** e la funzione **TAB**, come constatato nell'ultimo esempio considerato, per ottenere il movimento dell'immagine stessa sullo schermo.

L'impiego dell'istruzione **PRINT** con i comandi del cursore per creare ed animare figure, sebbene sia semplice, non è molto pratico in quanto

richiede numerosi passi di programma.

Inoltre, nei video-giochi è necessario che la figura che si muove sullo schermo possa reagire con altre figure presenti sullo schermo stesso; questo può essere ottenuto solamente con l'uso delle istruzioni **POKE** (immettere) e **PEEK** (leggere), le quali permettono di accedere direttamente alla memoria principale del computer.

Per potere impiegare queste istruzioni, occorre perciò conoscere le zone o aree di memoria RAM del computer destinate a contenere i caratteri ed i colori che si possono visualizzare sullo schermo.

Poiché tali zone di memoria sono diverse nel VIC-20 e nel C64, esse verranno esaminate separatamente. A seconda quindi del tipo di computer in Suo possesso (VIC-20 oppure C64), **dovrà fare riferimento al relativo capitolo che segue.**

4. - Grafica ed animazione con **POKE** e **PEEK** sul C64

Nella precedente lezione pratica ha constatato come si possano raffigurare sul video, mediante l'uso dell'istruzione **POKE**, diverse combinazioni dei colori del margine (bordo) e dello schermo (sfondo).

Ora accenneremo ad un'altra possibile utilizzazione dell'istruzione **POKE**: quella che consente di effettuare sul video vere e proprie animazioni a colori, movimenti di caratteri o figure, sulle mille possibili posizioni dello schermo.

In altre parole, possiamo immaginare lo schermo del C64 costituito da mille quadretti, come mostrato nella *fig. 11*, in ognuno dei quali può essere rappresentato uno qualsiasi dei diversi caratteri creati dal computer.

Occorre però precisare che ognuna delle mille posizioni dello schermo è funzionalmente messa in corrispondenza con due diverse zone o aree di memoria principale, riservate rispettivamente per immagazzinare i codici dei caratteri ed i codici dei colori.

Piú esattamente, **nella zona di memoria dello schermo del C64 che**

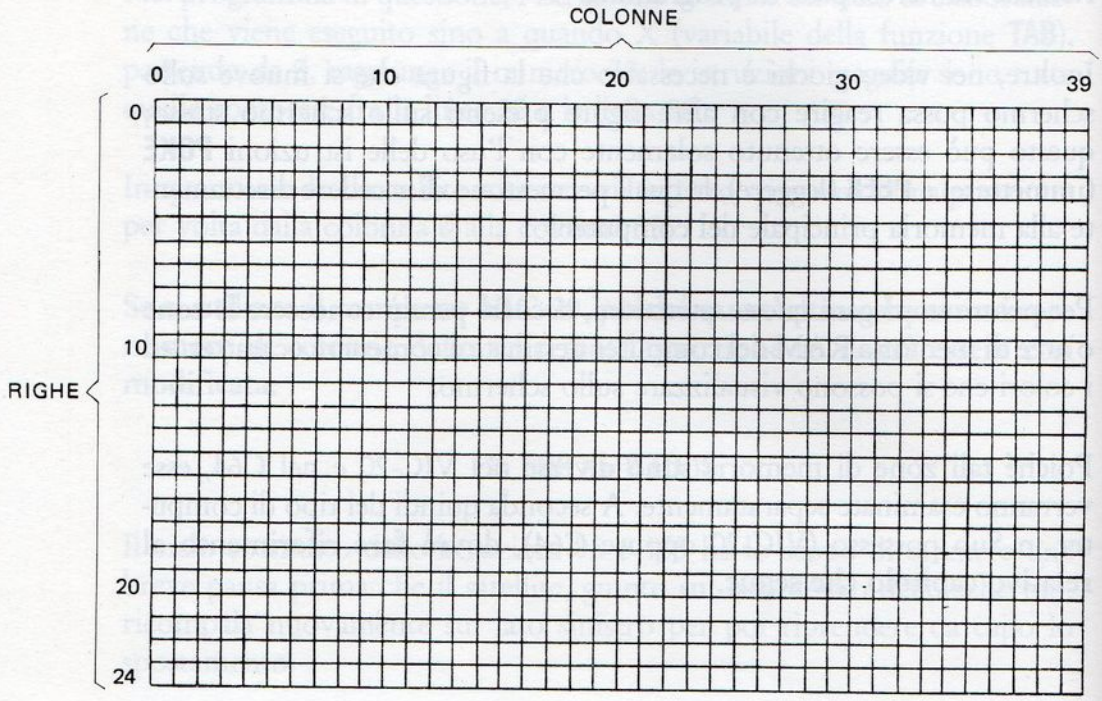


Fig. 11 – Schermo del C64.

inizia dalla locazione 1024 e termina alla locazione 2023 vengono conservati tutti i codici dei caratteri che si vogliono rappresentare in ognuna delle mille posizioni dello schermo.

Nella *fig. 12* è appunto riportata la rappresentazione grafica (mappa) delle locazioni di memoria dei caratteri.

Come può rilevare osservando tale figura, la locazione 1024 corrisponde all'angolo superiore sinistro dello schermo (riga Ø, colonna Ø), mentre la locazione 2023 corisponde all'angolo inferiore destro (riga 24, colonna 39).

La numerazione delle locazioni va quindi da sinistra a destra e dall'alto in basso, per cui l'ultima locazione della prima riga è 1063 e la prima locazione della seconda riga è 1064 e cosí via di seguito.

La zona di memoria dello schermo del C64, che inizia dalla locazione 55296 e termina con la locazione 56295, è invece riservata per immagazzinare i colori (espressi in codici numerici) che si possono utilizzare per ciascun carattere visualizzato sullo schermo.

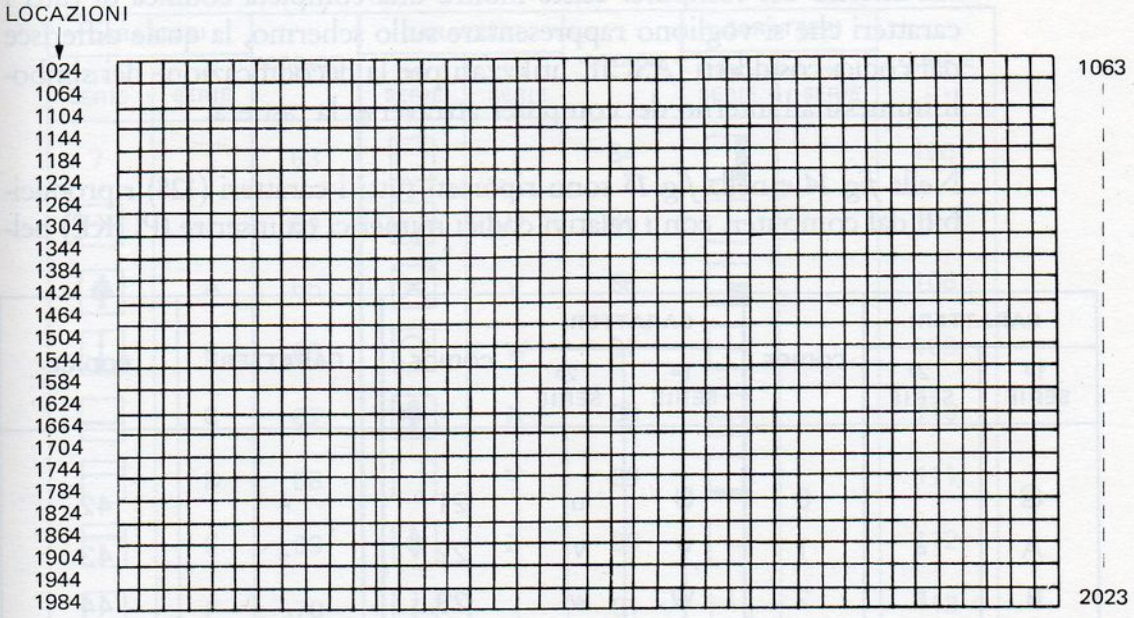


Fig. 12 – Mappa dei codici dei caratteri dello schermo del C64.

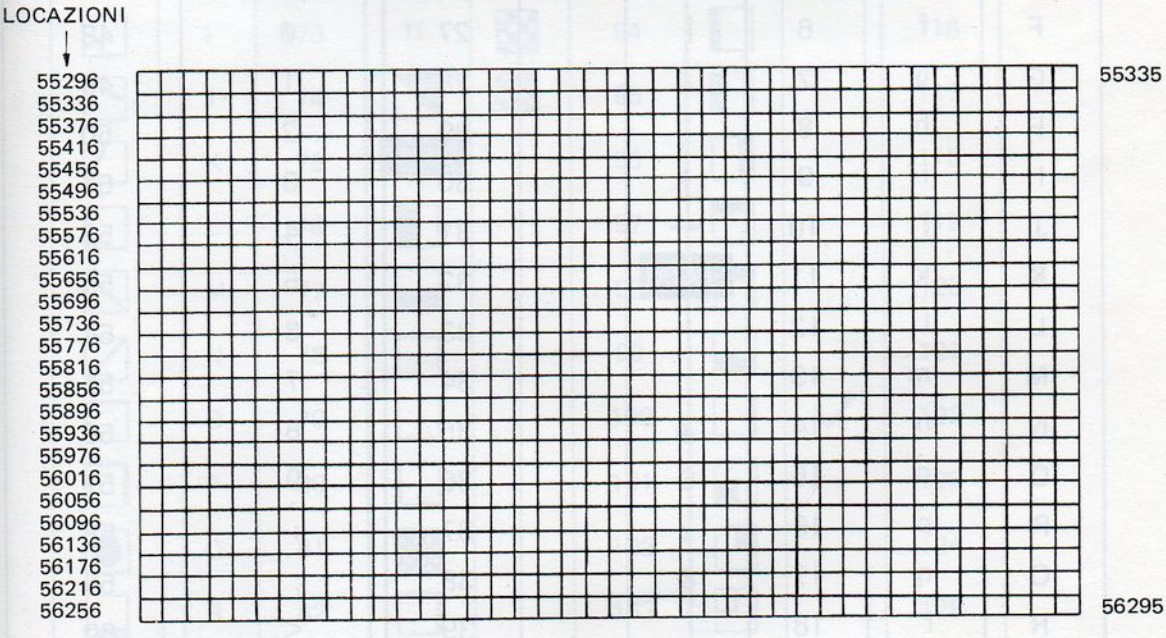


Fig. 13 – Mappa di memoria dei codici dei colori del C64.

La mappa di memoria dei colori è riportata nella *fig. 13*. Anche in questo caso la numerazione delle locazioni procede da sinistra a destra (partendo dalla locazione 55296) e dall'alto in basso.

All'interno del computer esiste inoltre una completa codifica di tutti i caratteri che si vogliono rappresentare sullo schermo, la quale differisce dai codici cosiddetti "ASCII" utilizzati per la decodificazione dei simboli immessi all'interno del computer attraverso la tastiera.

Nella *fig. 14* e nella *fig. 15* sono riportati tutti i caratteri (128) riproducibili dal computer, con i relativi codici numerici da inserire (POKE) nel-

CARATTERI		CODICE	CARATTERI		CODICE	CARATTERI	CODICE
1ª SERIE	2ª SERIE		1ª SERIE	2ª SERIE			
@		0	U	u	21	★	42
A	a	1	V	v	22	+	43
B	b	2	W	w	23	,	44
C	c	3	X	x	24	–	45
D	d	4	Y	y	25	.	46
E	e	5	Z	z	26	/	47
F	f	6	[		27	Ø	48
G	g	7	£		28	1	49
H	h	8	]		29	2	50
I	i	9	↑		30	3	51
J	j	10	←		31	4	52
K	k	11	SPAZIO		32	5	53
L	l	12	!		33	6	54
M	m	13	"		34	7	55
N	n	14	#		35	8	56
O	o	15	$		36	9	57
P	p	16	%		37	:	58
Q	q	17	&		38	;	59
R	r	18	'		39	<	60
S	s	19	(		40	=	61
T	t	20	)		41	>	62

Fig. 14 – Codici dei caratteri da rappresentare sullo schermo del C64.

CARATTERI 1ª SERIE	CARATTERI 2ª SERIE	CODICE
?		63
		64
	A	65
	B	66
	C	67
	D	68
	E	69
	F	70
	G	71
	H	72
	I	73
	J	74
	K	75
	L	76
	M	77
	N	78
	O	79
	P	80
	Q	81
	R	82
	S	83

CARATTERI 1ª SERIE	CARATTERI 2ª SERIE	CODICE
	T	84
	U	85
	V	86
	W	87
	X	88
	Y	89
	Z	90
		91
		92
		93
		94
		95
SPAZIO		96
		97
		98
		99
		100
		101
		102
		103
		104
		105

CARATTERI 1ª SERIE	CARATTERI 2ª SERIE	CODICE
		106
		107
		108
		109
		110
		111
		112
		113
		114
		115
		116
		117
		118
		119
		120
		121
	✓	122
		123
		124
		125
		126
		127

Fig. 15 – Codici dei caratteri da rappresentare sullo schermo del C64.

la memoria dello schermo (locazioni da 1024 a 2023) per poterli visualizzare.

Come può rilevare, alcuni gruppi di codici sono ripetuti per le lettere minuscole dell'alfabeto (2ª serie di caratteri).

Il passaggio dalle lettere maiuscole a quelle minuscole, come è noto, viene effettuato semplicemente pigiando simultaneamente i due tasti di controllo SHIFT e C= .

Le due serie di caratteri sono disponibili solamente una per volta.

I caratteri riportati nella *fig. 14* e nella *fig. 15* sono anche riproducibili in negativo (o video-inverso). Ciò si ottiene addizionando il numero 128 al valore di codice del carattere che si vuole riprodurre.

Pertanto, i caratteri in negativo hanno i numeri di codice che vanno da 128 a 255.

Per esempio, se si vuole rappresentare la lettera M, il cui codice è 13, in negativo, si deve assegnare ad essa il codice 141 (ossia 13 + 128 = 141).

Osservando ancora la *fig. 14* e la *fig. 15* si può rilevare che anche lo spazio ha un numero di codice (anzi due), e questo perché una locazione caratteri non può essere considerata vuota.

Nello specchietto che segue sono riportati i codici numerici da inserire (POKE) nelle locazioni della memoria per il colore (locazioni da 55296 a 56295) per poter colorare il carattere corrispondente.

Codice	Colore	Codice	Colore
Ø	nero	8	arancio
1	bianco	9	marrone
2	rosso	1Ø	rosso chiaro
3	blu-verde	11	grigio 1
4	porpora	12	grigio 2
5	verde	13	verde chiaro
6	blu	14	azzurro
7	giallo	15	grigio 3

Dopo queste precisazioni, possiamo passare alla parte effettivamente operativa della visualizzazione dei caratteri sullo schermo.

Come primo esempio supponiamo di volere riportare il simbolo del "cuore" (♥), corrispondente al numero di codice 83 (*fig 15*), al centro dello schermo (locazione memoria caratteri 1524).

Possiamo quindi scrivere l'istruzione BASIC ad esecuzione immediata che segue:

POKE 1524,83

locazione — codice carattere

Sullo schermo apparirà la figura di un "cuore" avente lo stesso colore standard dei caratteri del C64, ossia azzurro. Constaterà cosí che il carattere in questione è stato direttamente inserito nella memoria del computer senza utilizzare lo spostamento del cursore.

Se ora volessimo fare in modo che il "cuore" diventi di colore rosso, dovremmo battere la seguente altra istruzione:

POKE 55796,2

locazione — codice colore

dove la locazione 55796 corrisponde al centro dello schermo per quanto riguarda la mappa di memoria dei colori, ed il numero 2 è il codice del colore rosso.

Per cancellare il "cuore" dalla casella centrale dello schermo sarà sufficiente ricoprirlo con uno spazio (codice carattere 32).

Scrivendo quindi:

POKE 1524,32

il cuore sparirà dallo schermo.

Per visualizzare invece, sempre al centro dello schermo, un "cuore" che cambia continuamente colore, Lei può eseguire il breve programma che segue:

```
10 PRINT"■"
20 FOR X=0 TO15
30 POKE 1524,83
40 POKE 55796,X
50 FOR T=1 TO 500:NEXT T
60 NEXT X
70 GOTO 10
80 END
```

Come può rilevare, in questo programma l'istruzione 2Ø imposta un'iterazione che fornisce alla variabile X i valori compresi tra Ø e 15 (corrispondenti al codice dei colori) da inserire (POKE) nella locazione per la memoria del colore 55796.

Il ciclo di ritardo (istruzione 5Ø) consente di percepire il cambiamento dei colori. Infatti, senza questa istruzione i colori cambierebbero cosí velocemente da non poter essere individuati.

Riassumendo quanto sinora precisato, per poter rappresentare un dato carattere in una delle possibili posizioni dello schermo è necessario conoscere i seguenti dati:

- **la locazione della memoria caratteri (da 1024 a 2023) e la locazione della memoria colori (da 55296 a 56295), corrispondenti alla posizione dello schermo in cui si desidera visualizzare il carattere;**
- **il codice del carattere (da Ø a 127) e quello del colore (da Ø a 15) prescelti.**

Vediamo ora come occorre procedere per determinare in quale posizione dello schermo si deve visualizzare un dato carattere con il relativo colore.

Le formule per calcolare queste posizioni sono:

```
P=1024+X+40*Y    (posizione carattere)
C=55296+X+40*Y   (posizione colore)
```

dove X è il numero della colonna e Y è il numero della riga che individuano sullo schermo la posizione in cui si vuole rappresentare un determinato carattere.

Ripulisca lo schermo cancellando tutte le scritte.

Ora supponga, per esempio, di volere visualizzare la lettera A di colore giallo nella colonna 10 (undicesima colonna) della riga 15 (sedicesima riga).

In primo luogo deve calcolare le due corrispondenti locazioni della memoria caratteri e della memoria colore, che risultano:

```
P=  1024+10+40*15=1634
C=55296+10+40*15=55906
```

Poiché il codice carattere della lettera A è 1 ed il codice del colore giallo è 7, è sufficiente battere le due istruzioni che seguono:

```
POKE 1634,1
POKE 55906,7
```

per vedere comparire sullo schermo, in corrispondenza della colonna 10 e della riga 15, la lettera A di colore giallo.

Seguendo il sistema descritto, Lei può esercitarsi a visualizzare caratteri con diversi colori nelle varie posizioni dello schermo.

Per vedere, invece, muovere una pallina (codice 81) **verticalmente** dall'alto in basso e da sinistra a destra sullo schermo, occorre scrivere ed eseguire (con il comando **RUN**) il semplice programma che segue:

```
10 PRINT"□"
15 FOR X=0 TO 39
20 FOR Y=0 TO 24
30 P=1024+X+40*Y
40 POKE P,81
50 NEXT Y
60 NEXT X
65 GOTO 10
70 END
```

Con questo programma il simbolo grafico della pallina viene raffigurato sullo schermo a partire dalla posizione piú in alto a sinistra (locazione 1024) e quindi, di colonna in colonna, in tutte le altre posizioni dello schermo stesso (che, come è noto, sono mille).

L'istruzione 3Ø provvede a calcolare la locazione di memoria per il codice di carattere 81 (quello della pallina) con X variabile da Ø a 39 (colonne) ed Y variabile da Ø a 24 (righe).

I valori X e Y sono determinati rispettivamente con i due cicli **FOR**, riportati nelle linee di programma 15 e 2Ø.

Con l'istruzione 4Ø viene inserito (POKE) il codice del carattere 81 nell'apposita locazione di memoria calcolata con l'istruzione 3Ø.

Se si desidera fare muovere la solita pallina (codice 81) in modo che si sposti **orizzontalmente** da sinistra a destra e dall'alto in basso dello schermo, si devono modificare le istruzioni 15, 2Ø, 3Ø del precedente pro-

gramma, come di seguito indicato:

```
15 FOR X=0 TO 24
20 FOR Y=0 TO 39
30 P=1024+Y+40*X
```

Infine, per fare in modo che la pallina, dopo avere raggiunto la mezzeria centrale dello schermo (che corrisponde alla locazione 1543), ritorni in "retromarcia" alla posizione di partenza (angolo superiore sinistro corrispondente alla locazione 1024), occorre ampliare e modificare il precedente programma, come indicato nel listato della *fig. 16*.

```
10 REM**PALLINA**
15 PRINT"■"
20 GOTO 55
25 PRINT"■"
30 L1=12
40 L2=0
45 P1=-1
46 Y1=39
47 Y2=0
50 GOTO 70
55 L1=0
60 L2=12
65 P1=1
66 Y1=0
67 Y2=39
70 FOR X=L1 TO L2 STEP P1
75 FOR Y=Y1 TO Y2 STEP P1
80 POKE 1024+Y+40*X,81
83 FOR T=1 TO 10:NEXT
85 NEXT Y
90 NEXT X
95 IF L1=0 THEN 25
100 PRINT"■":GOTO 55
102 END
```

Fig. 16 – Listato del programma (per C64) il quale visualizza sullo schermo una pallina che si muove dall'alto in basso sino a portarsi al centro dello schermo e quindi ritorna indietro al punto di partenza.

In tale listato le istruzioni 30÷47 stabiliscono i punti di partenza (riga 12, colonna 39) e di arrivo (riga 0, colonna 0) della pallina durante il ritorno, mentre le istruzioni 55÷67 determinano i punti di partenza (riga 0, colonna 0) e di arrivo (riga 12, colonna 39) della stessa pallina durante l'andata.

Alle linee di programma 70 e 75 si hanno due istruzioni che agiscono, se

cosí si può dire, in due direzioni: una prima volta consentono alla pallina di muoversi verso destra dalla posizione iniziale dello schermo (indirizzo di memoria 1024) fino a portarsi nella posizione centrale (corrispondente all'indirizzo 1543); una seconda volta permettono alla pallina di muoversi in senso opposto, cioè dal punto centrale dello schermo fino all'angolo superiore sinistro dello stesso (*fig. 17*).

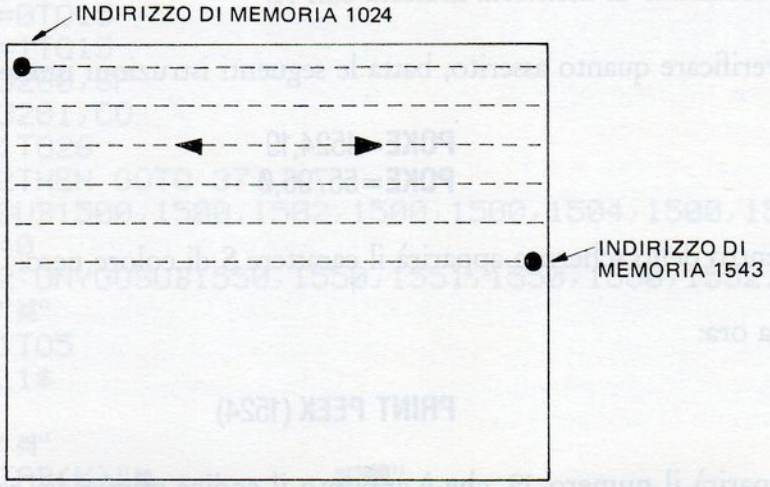


Fig. 17 - Spostamento della pallina sullo schermo del C64 ottenuto con il programma listato nella fig. 16.

L'istruzione 95 si chiede se L1=Ø, ossia se il limite inferiore per il controllo della riga è Ø: ciò equivale a domandarsi se la direzione del movimento della pallina è stata verso destra; quindi occorre subito farle assumere il movimento opposto (verso sinistra).

Il ciclo di "ritardo" previsto dall'istruzione 83 consente, a chi osserva lo schermo, di rendersi conto visivamente del movimento (verso destra e verso sinistra) della pallina.

A questo punto può immettere il programma descritto nel computer ed eseguirlo.

La pallina che comparirà sullo schermo può anche essere colorata, per esempio apparire gialla. Per ottenere ciò è sufficiente inserire nel listato del programma della *fig. 16* la seguente istruzione:

```
81 POKE 55296+Y+40*X,7
```

Cambiando il codice numerico del colore (7), si può rappresentare la pallina in uno degli altri quindici colori previsti dal computer.

Verificato il regolare funzionamento del programma in questione, lo cancelli dalla memoria principale del computer e ripulisca lo schermo mediante gli appositi comandi già piú volte usati.

Come già specificato, l'istruzione **POKE** viene utilizzata spesso con la funzione **PEEK** (X), che consente di "sbirciare" ossia leggere il contenuto della locazione di memoria indicata con X.

Per verificare quanto asserito, batta le seguenti istruzioni immediate:

```
POKE=1524,19
POKE=55796,Ø
```

Al centro dello schermo apparirà il carattere S di colore nero.

Scriva ora:

```
PRINT PEEK (1524)
```

Comparirà il numero 19, che è appunto il codice numerico, relativo al carattere S (*fig. 14*), contenuto nella locazione di memoria 1524.

ANIMAZIONE E COLORI 2

Eseguiremo ora una seconda versione del programma denominato "*Animazione e colori*", realizzato nella precedente lezione, nel quale verrà ora utilizzata l'istruzione **POKE** per creare l'effetto di illuminazione delle finestre delle case ad ogni passaggio dell'aereo.

Il listato del nuovo programma è riportato nella *fig. 18* e nella *fig. 19*; esaminiamolo limitatamente alle nuove istruzioni che sono state aggiunte.

Nell'istruzione 35 si ha il salto alle subroutine (GOSUB) aventi inizio al numero di linea corrispondente al valore di X, variabile da 1 a 12.

In altre parole, quando X=1, si ha il salto al primo numero di linea (1500) relativo all'istruzione **RETURN**; quando X=2, si ha il salto al secondo numero di linea (ancora 1500); allorché X=3, si ha il salto al numero di linea 15Ø2, prima subroutine, e cosí via, sino a quando X=12 per cui si ha il salto al numero di linea 15Ø8 (ultima subroutine).

```
10 REM*ANIMAZIONE E COLORI 2*
12 PRINT"■"
20 GOSUB1000
21 L1$="                                                  "
22 POKE53280,1
24 FORSF=0TO15
26 FORCO=1TO15
27 POKE53280,SF
28 POKE53281,CO
30 FORX=1TO26
31 IFX>12THEN GOTO 37
35 ONXGOSUB1500,1500,1502,1500,1500,1504,1500,1500,1506,1500,1500,1508
36 GOTO 40
37 Y=X-12:ONYGOSUB1550,1550,1551,1550,1550,1552,1550,1553,1550,1554
40 PRINT"■"
42 FORY=1TO5
45 PRINTL1$
47 NEXTY
48 PRINT"■"
55 PRINTTAB(X)"■          ■"
60 PRINTTAB(X)"■┌──────┐  ■"
70 PRINTTAB(X)"■│ BASIC ├─■"
80 PRINTTAB(X)"■└──────┘  ■"
90 PRINTTAB(X)"■          ■"
91 FORZ=1TO50:NEXTZ
95 IFX=26 THENGOSUB1550
100 NEXTX
130 NEXTCO
136 NEXTSF
140 PRINT"■":END
1000 PRINT"■■■■■■■■■■■■"
1010 PRINT"■          ■  ■  ■  ■"
1020 PRINT"■          ■  ■  ■  ■"
1030 PRINT"■    ■  ■  ■  ■  ■  ■"
1040 PRINT"■    ■  ■  ■  ■  ■  ■"
1050 PRINT"■    ■  ■  ■  ■  ■  ■"
1060 PRINT"■    ■  ■  ■  ■  ■  ■"
1070 PRINT"■ ■     ■  ■  ■  ■  ■"
1080 PRINT"■ ■"
1090 PRINT"■ ■"
1100 PRINT"■ ■"
1110 PRINT"■ ■"
```

Fig. 18 – Listato del programma "Animazione e colori 2" per il C64.

```
1120 RETURN
1500 RETURN
1502 POKE55901,7:POKE55987,7:POKE55973,7:RETURN
1504 POKE55952,7:POKE55999,7:POKE55883,7:RETURN
1506 POKE55835,7:POKE56048,7:POKE56058,7:RETURN
1508 POKE55928,7:POKE56035,7:POKE55827,7:RETURN
1550 RETURN
1551 POKE55901,0:POKE55987,0:POKE55973,0:RETURN
1552 POKE55952,0:POKE55999,0:POKE55883,0:RETURN
1553 POKE55835,0:POKE56048,0:POKE56058,0:RETURN
1554 POKE55928,0:POKE56035,0:POKE55827,0:RETURN
```

Fig. 19 – Proseguimento del listato del programma "Animazione e colori 2" per il C64.

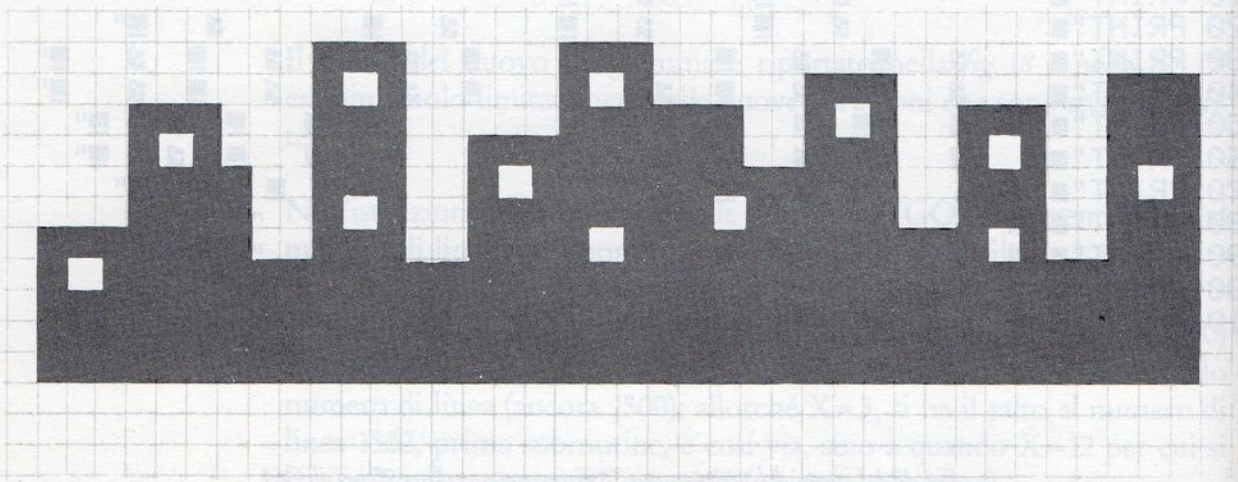
Le subroutine 15Ø2, 15Ø4, 15Ø6 e 15Ø8 comprendono ognuna tre istruzioni, ciascuna delle quali inserisce (POKE) il codice colore 7 (giallo) nella corrispondente locazione di memoria, creando cosí l'illusione che le finestre delle case (la cui posizione è indicata sempre da tali locazioni) si illuminino al passaggio dell'aereo.

Al termine di ogni subroutine vi è l'istruzione **RETURN**, che ordina il ritorno al programma principale.

La posizione delle finestre illuminate è mostrata nella *fig. 20*.

Quando l'aereo giunge in prossimità del lato destro dello schermo, le

Fig. 20 – Posizione delle "finestre" illuminate, relative al programma della fig. 18 e della fig. 19 (C64).



finestre si oscurano e ciò viene ottenuto con l'istruzione 37, nella quale si ha il salto alle subroutine aventi inizio al numero di linea specificato dal valore della variabile Y.

Questo secondo gruppo di subroutine è costituito dalle istruzioni 1551, 1552, 1553 e 1554, ognuna delle quali è formata da tre **POKE**, aventi le stesse locazioni di quelle riportate nelle precedenti subroutine (15Ø2, 15Ø4, 15Ø6 e 15Ø8); ma in questo caso il codice del colore inserito è Ø corrispondente al nero, per creare l'effetto di oscuramento delle finestre.

La concomitanza fra lo spostamento dell'aereo e l'illuminazione delle finestre si verifica per il fatto che la variabile X, oltre a stabilire la colonna in cui viene posizionato l'aereo durante lo spostamento, è anche utilizzata (come si è visto nell'istruzione 35) per ottenere il salto alle subroutine.

Esaminate le modifiche apportate al programma, può ora passare ad eseguirlo, tenendo presente quanto precisato nella precedente lezione a proposito delle istruzioni relative alla realizzazione dell'aereo (istruzioni 5Ø÷9Ø) e delle case (istruzioni 1Ø1Ø÷1Ø95).

Eseguendo il programma con il comando **RUN**, vedrà appunto illuminarsi sullo schermo le finestre delle case durante il passaggio dell'aereo.

A conclusione della lezione, occorre ancora precisare che se si compiono errori inserendo (POKE) valori in locazioni di memoria errate oppure inserendo valori errati in locazioni corrette, il funzionamento del computer può alterarsi.

Ciò però non deve assolutamente preoccupare, in quanto è possibile ripristinare le normali condizioni di funzionamento premendo contemporaneamente i tasti RUN | STOP e RESTORE oppure, nei casi piú gravi, spegnendo e riaccendendo, dopo almeno 5÷6 secondi, il computer.

5. - Grafica ed animazione con POKE e PEEK sul VIC-20

Nella precedente lezione pratica ha constatato come si possano raffigurare sul video, mediante l'uso dell'istruzione **POKE**, diverse combinazioni dei colori del margine (bordo) e dello schermo (sfondo).

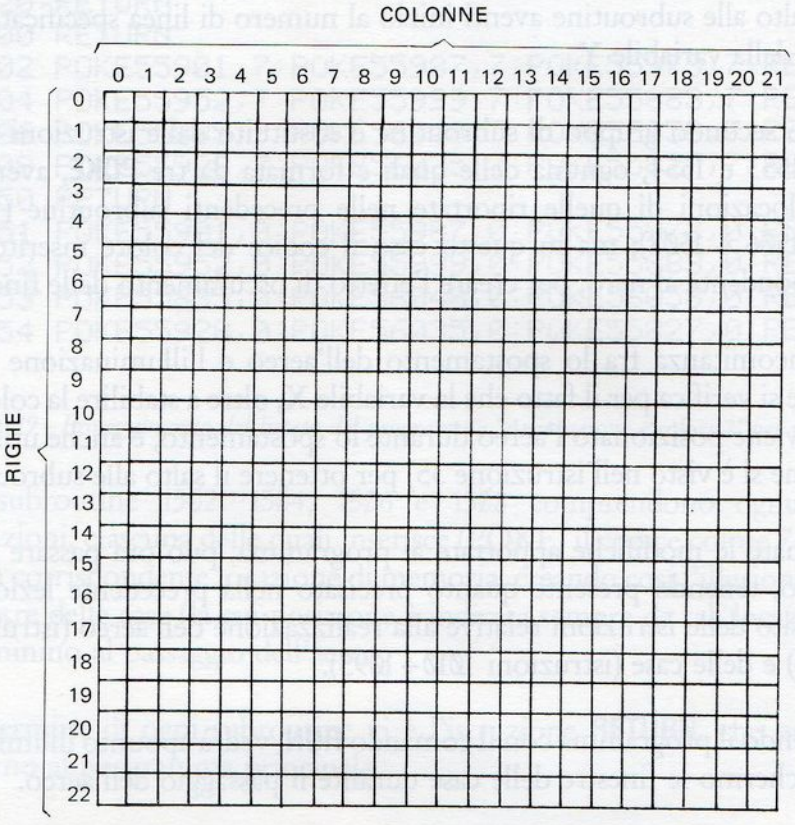


Fig. 21 – Schermo del VIC-20.

Ora vedremo un'altra possibile utilizzazione dell'istruzione **POKE**: quella che consente di effettuare sul video vere e proprie animazioni, movimenti di caratteri o figure, sulle 506 possibili posizioni dello schermo, eventualmente colorando questi ultimi.

In altre parole, possiamo immaginare lo schermo del VIC-20 costituito da 506 quadretti, come mostrato nella *fig. 21*, in ognuno dei quali può essere rappresentato uno qualsiasi dei diversi caratteri creati dal computer.

Occorre però precisare che ognuna delle 506 posizioni dello schermo del VIC-20 è funzionalmente messa in corrispondenza con due diverse zone di memoria principale, riservate rispettivamente per immagazzinare i codici dei caratteri ed i codici dei colori.

Piú esattamente, **nella zona di memoria dello schermo del VIC-20, che inizia dalla locazione 7680 e termina alla locazione 8185, sono conservati tutti i codici dei caratteri** che si vogliono rappresentare in ognuna

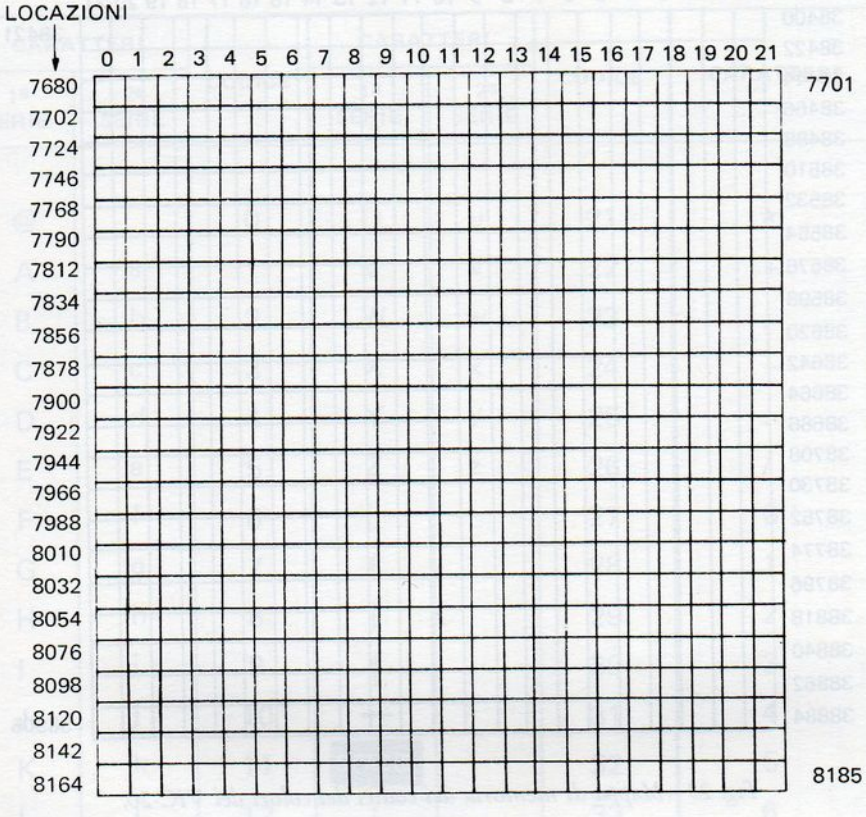


Fig. 22 – Mappa dei codici dei caratteri dello schermo del VIC-20.

delle 506 posizioni dello schermo stesso.

Nella *fig. 22* è appunto riportata la rappresentazione grafica (mappa) delle locazioni di memoria dei caratteri.

Come si può rilevare osservando tale figura, la locazione 7680 corrisponde all'angolo superiore sinistro dello schermo (riga Ø, colonna Ø), mentre la locazione 8185 corrisponde all'angolo inferiore destro (riga 22, colonna 21).

La numerazione delle locazioni va quindi da sinistra a destra e dall'alto in basso, per cui l'ultima locazione della prima riga (colonna 21) è 7701, la prima locazione della seconda riga è 7702 e cosí via di seguito.

La zona di memoria dello schermo del VIC-20, che inizia dalla locazione 38400 e termina con la locazione 38905, è invece riservata per immagazzinare i colori (espressi in codici numerici) che si possono utilizzare per ciascun carattere visualizzato sullo schermo.

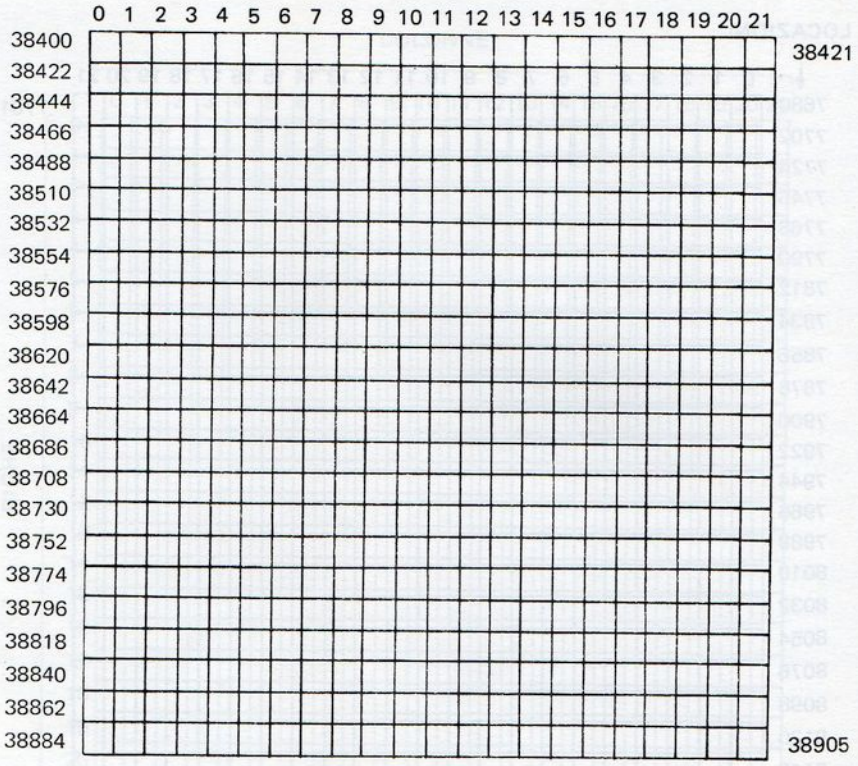


Fig. 23 – Mappa di memoria dei codici dei colori del VIC-20.

La mappa di memoria dei colori è riportata nella *fig. 23*. Anche in questo caso la numerazione delle locazioni procede da sinistra a destra (partendo dalla locazione 38400) e dall'alto in basso.

All'interno del computer esiste inoltre una completa codifica di tutti i caratteri che si vogliono rappresentare sullo schermo, codifica che differisce dai codici cosiddetti "ASCII" che servono alla decodificazione dei simboli che si immettono nel computer attraverso la tastiera.

Nella *fig. 24* e nella *fig. 25* sono riportati tutti i caratteri (128) riproducibili dal computer, con i relativi codici numerici da inserire (POKE) nella memoria dello schermo (locazioni da 7680 a 8185) per poterli visualizzare.

Come può rilevare, alcuni gruppi di codici sono ripetuti per le lettere minuscole dell'alfabeto (2ª serie di caratteri).

Il passaggio dalle lettere maiuscole a quelle minuscole, come è noto, viene effettuato semplicemente pigiando simultaneamente i due tasti di controllo SHIFT e C= .

CARATTERI		CODICE	CARATTERI		CODICE	CARATTERI	CODICE
1ª SERIE	2ª SERIE		1ª SERIE	2ª SERIE			
@		0	U	u	21	★	42
A	a	1	V	v	22	+	43
B	b	2	W	w	23	,	44
C	c	3	X	x	24	–	45
D	d	4	Y	y	25	.	46
E	e	5	Z	z	26	/	47
F	f	6	[		27	Ø	48
G	g	7	£		28	1	49
H	h	8	]		29	2	50
I	i	9	↑		30	3	51
J	j	10	←		31	4	52
K	k	11	SPAZIO		32	5	53
L	l	12	!		33	6	54
M	m	13	"		34	7	55
N	n	14	#		35	8	56
O	o	15	$		36	9	57
P	p	16	%		37	:	58
Q	q	17	&		38	;	59
R	r	18	'		39	<	60
S	s	19	(		40	=	61
T	t	20	)		41	>	62

Fig. 24 – Codici dei caratteri da rappresentare sullo schermo del VIC-20.

Le due serie di caratteri sono disponibili solamente una per volta.

I caratteri riportati nella *fig. 24* e nella *fig. 25* sono anche riproducibili in negativo (o video-inverso). Ciò si ottiene addizionando il numero 128 al valore di codice del carattere che si vuole riprodurre. Pertanto i caratteri in negativo hanno i numeri di codici che vanno da 128 a 255.

CARATTERI 1ª SERIE	CARATTERI 2ª SERIE	CODICE
?		63
		64
♠	A	65
	B	66
	C	67
	D	68
	E	69
	F	70
	G	71
	H	72
	I	73
	J	74
	K	75
	L	76
	M	77
	N	78
	O	79
	P	80
●	Q	81
	R	82
♥	S	83

CARATTERI 1ª SERIE	CARATTERI 2ª SERIE	CODICE
	T	84
	U	85
	V	86
○	W	87
♣	X	88
	Y	89
♦	Z	90
		91
		92
		93
π		94
		95
SPAZIO		96
		97
		98
		99
		100
		101
		102
		103
		104
		105

CARATTERI 1ª SERIE	CARATTERI 2ª SERIE	CODICE
		106
		107
		108
		109
		110
		111
		112
		113
		114
		115
		116
		117
		118
		119
		120
		121
	✓	122
		123
		124
		125
		126
		127

Fig. 25 – Codici dei caratteri da rappresentare sullo schermo del VIC-20.

Per esempio, se si vuole rappresentare la lettera M, il cui codice è 13, in negativo, si deve assegnare ad essa il codice 141 (ossia 13 + 128 = 141).

Osservando ancora la *fig. 24* e la *fig. 25* si può rilevare che anche lo spazio ha un numero di codice (anzi due) e questo perché una locazione caratteri non può essere considerata vuota.

Nello specchietto che segue sono riportati i codici numerici, da inserire (POKE) nelle locazioni della memoria per il colore (locazioni da 38400 a 38905) per poter colorare il carattere corrispondente.

Codice	Colore
Ø	nero
1	bianco
2	rosso
3	blu-verde
4	porpora
5	verde
6	blu
7	giallo

Dopo queste precisazioni, possiamo passare alla parte effettivamente operativa della visualizzazione dei caratteri sullo schermo.

Come esempio supponiamo di voler riportare il simbolo del "cuore" (♥) circa al centro dello schermo visualizzandolo nel colore rosso (codice 2).

Il simbolo ♥ , come si vede dalla tabella dei codici (*fig. 25*), è contraddistinto con il numero 83. La posizione prescelta corrisponde, per quanto riguarda i caratteri, alla locazione 7910 e, per i colori, alla locazione 38630. Possiamo quindi scrivere le due istruzioni BASIC ad esecuzione immediata seguenti:

```
POKE 7910,83
```

locazione — codice carattere

```
POKE 38630,2
```

locazione — codice colore

In questo modo il carattere (♥) viene direttamente inserito nella memoria del computer, senza utilizzare lo spostamento del cursore.

Se volessimo cambiare subito il colore del cuore, cioè farlo apparire per esempio blu-verde (colore che corrisponde al tasto CYN ed al codice numerico 3) anziché rosso, sempre mantenendo il fondo bianco dello schermo, basterebbe scrivere:

POKE 38630,3

Per cancellare il "cuore" dalla casella centrale dello schermo sarà sufficiente ricoprirlo con uno spazio bianco (codice carattere 32): infatti, uno spazio bianco su fondo bianco produce l'effetto di una cancellatura.

Scrivendo quindi:

POKE 7910,32

il "cuore" sparirà dallo schermo.

Per visualizzare, sempre al centro dello schermo, un "cuore" che cambia continuamente colore, Lei può eseguire il breve programma che segue:

```
10 PRINT"□"
20 FOR X=0 TO 7
30 POKE 7910,83
40 POKE 38630,X
50 FOR T=1 TO 500:NEXT T
60 NEXT X
70 GOTO 10
80 END
```

Come può rilevare, in questo programma l'istruzione 2Ø imposta un'iterazione che fornisce alla variabile X i valori da Ø a 7 (corrispondenti al codice dei colori) da inserire nella locazione per la memoria del colore 3863Ø.

Il ciclo di ritardo (istruzione 5Ø) consente di percepire il cambiamento dei colori. Infatti, senza questa istruzione i colori cambierebbero cosí velocemente da non poter essere individuati.

Riassumendo quanto sinora precisato, per potere rappresentare un determinato carattere in una delle 506 possibili posizioni dello schermo è necessario conoscere i seguenti dati:

- **la locazione della memoria caratteri (da 7680 a 8185) e la locazione della memoria colori (da 38400 a 38905) corrispondenti alla posizione dello schermo in cui si desidera visualizzare il carattere;**

- **il codice del carattere (da Ø a 127) e quello del colore (da Ø a 7) prescelti.**

Vediamo ora come occorre procedere per determinare in quale posizione dello schermo si deve visualizzare un dato carattere con il relativo colore.

Le formule per calcolare queste posizioni sono:

```
P=7680+X+22*Y    (posizione carattere)
C=38400+X+22*Y   (posizione colore)
```

dove X è il numero delle colonne e Y è il numero delle righe che individuano sullo schermo la posizione in cui si vuole rappresentare un determinato carattere.

Ripulisca lo schermo cancellando tutte le scritte.

Supponga, per esempio, di voler visualizzare la lettera A di colore verde nella colonna 1Ø (undicesima colonna) della riga 15 (sedicesima riga).

In primo luogo deve calcolare le due corrispondenti locazioni della memoria caratteri e della memoria colore, che risultano:

```
P=7680+10+22*15=8020
C=38400+10+22*15=38740
```

Poiché il codice carattere della lettera A è 1 ed il codice del colore verde è 5, è sufficiente battere le due istruzioni che seguono:

```
POKE 8020,1
POKE 38740,5
```

per vedere comparire sullo schermo, in corrispondenza della colonna 10 e della riga 15, la lettera A di colore verde.

Seguendo il sistema descritto, Lei può esercitarsi a visualizzare caratteri con diversi colori, nelle varie posizioni dello schermo.

A questo riguardo tenga presente che, per potere visualizzare qualsiasi carattere su schermo bianco, il carattere deve sempre essere colorato.

Per vedere, invece, muovere una pallina (codice 81) **verticalmente** dall'alto in basso e da sinistra a destra dello schermo, occorre scrivere ed eseguire (con il comando **RUN**) il semplice programma che segue:

```
10 PRINT"□"
15 FOR X=0 TO 21
20 FOR Y=0 TO 22
25 POKE 36879,8
30 P=7680+X+22*Y
40 POKE P,81
50 NEXT Y
60 NEXT X
65 GOTO 10
70 END
```

Con questo programma il simbolo grafico della pallina viene raffigurato sullo schermo a partire dalla posizione piú in alto a sinistra (locazione 7680) e quindi, di colonna in colonna, in tutte le altre posizioni.

Siccome la pallina è di colore bianco, essa non si noterebbe sullo schermo anch'esso bianco; perciò quest'ultimo viene oscurato con l'istruzione 25, in modo da rendere visibile il movimento della pallina stessa.

L'istruzione 3Ø provvede a calcolare la locazione di memoria per il codice di carattere 81 (quello della pallina) con X variabile da Ø a 21 (colonne) ed Y variabile da Ø a 22 (righe).

I valori X e Y sono determinati rispettivamente con i cicli FOR delle linee di programma 15 e 2Ø.

Con l'istruzione 4Ø viene inserito (POKE) il codice del carattere 81 nell'apposita locazione di memoria calcolata con l'istruzione 3Ø.

Se si desidera fare muovere la solita pallina (codice 81) in modo che si sposti **orizzontalmente** da sinistra a destra e dall'alto in basso dello schermo, si devono modificare le istruzioni 15, 2Ø, 3Ø del precedente programma, come di seguito indicato:

```
15 FOR X=Ø TO 22
2Ø FOR Y=Ø TO 21
3Ø P=768Ø+Y+22*X
```

Infine, per fare in modo che la pallina, dopo avere raggiunto la mezzeria centrale dello schermo (che corrisponde alla locazione 7943), ritorni in "retromarcia" alla posizione di partenza (angolo superiore sinistro corri-

spondente alla locazione 7680), occorre ampliare e modificare il precedente programma come indicato nel listato della *fig. 26*.

```
10 REM**PALLINA**
15 PRINT"□"
20 GOTO 55
25 PRINT"□"
30 L1=11
40 L2=0
45 P1=-1
46 Y1=21
47 Y2=0
50 GOTO 70
55 POKE 36879,8
57 L1=0
60 L2=11
65 P1=1
66 Y1=0
67 Y2=21
70 FOR X=L1 TO L2 STEP P1
75 FOR Y=Y1 TO Y2 STEP P1
80 POKE 7680+Y+22*X,81
83 FOR T=1 TO 10:NEXT
85 NEXT Y
90 NEXT X
95 IF L1=0 THEN 25
100 PRINT"□":GOTO 55
102 END
```

Fig. 26 – Listato del programma (per VIC-20) il quale visualizza sullo schermo una pallina che si sposta dall'alto in basso sino a portarsi al centro dello schermo e quindi ritorna indietro al punto di partenza.

In tale listato le istruzioni 30÷47 stabiliscono i punti di partenza (riga 11, colonna 21) e di arrivo (riga 0, colonna 0) della pallina durante il ritorno, mentre le istruzioni 57÷67 determinano i punti di partenza (riga 0, colonna 0) e di arrivo (riga 11, colonna 21) della stessa pallina durante l'andata.

Alle linee di programma 70 e 75 si hanno due istruzioni FOR che agiscono, se cosí si può dire, in due direzioni: una prima volta consentono alla pallina di muoversi verso destra dalla posizione iniziale dello schermo (indirizzo di memoria 7680) fino a portarsi nella posizione centrale (corrispondente all'indirizzo 7943); una seconda volta permettono alla pallina di muoversi in senso opposto, cioè dalla mezzeria centrale dello schermo all'angolo superiore sinistro dello stesso (*fig. 27*).

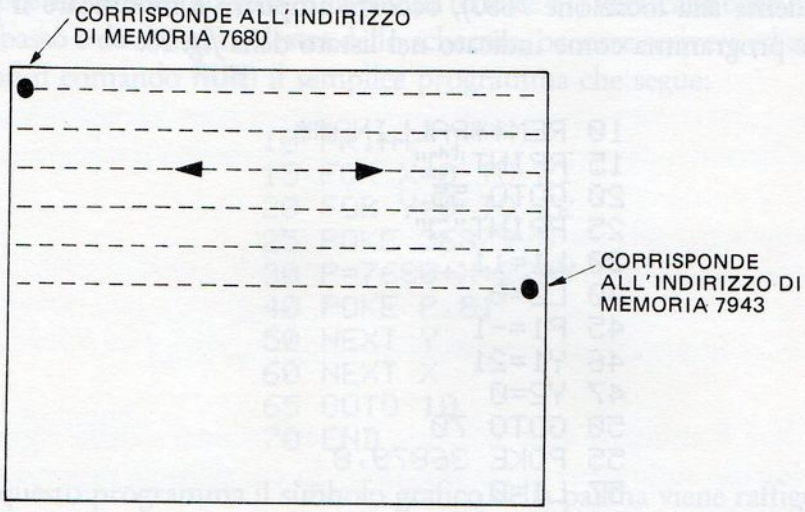


Fig. 27 – Spostamento della pallina sullo schermo del VIC-20 ottenuto con il programma listato nella fig. 26.

L'istruzione 95 si chiede se L1=Ø, ossia se il limite inferiore per il controllo della riga è Ø: ciò equivale a domandarsi se la direzione del movimento della pallina è stata verso destra; quindi, occorre subito farle assumere il movimento opposto (verso sinistra).

Il ciclo di "ritardo" previsto dall'istruzione 83 consente, a chi osserva lo schermo, di rendersi conto visivamente del movimento (verso destra e verso sinistra) della pallina.

A questo punto può immettere il programma descritto nel computer ed eseguirlo.

La pallina che comparirà sullo schermo può anche essere colorata, per esempio apparire gialla. Per ottenere ciò è sufficiente inserire nel listato del programma della *fig. 26* la seguente istruzione:

```
81 POKE 38400+Y+22*X,7
```

Cambiando il codice numerico del colore (7), si può rappresentare la pallina in uno degli altri sette colori previsti dal computer.

Verificato il regolare funzionamento del programma in questione, lo cancelli dalla memoria principale del computer e ripulisca lo schermo mediante gli appositi comandi già piú volte usati.

Come già specificato, l'istruzione **POKE** viene utilizzata spesso con la funzione **PEEK** (X), che consente di "sbirciare" ossia leggere il contenuto della locazione di memoria indicata con X.

Per verificare quanto asserito, batta le seguenti istruzioni immediate:

```
POKE 7910,19
POKE 38630,0
```

Al centro dello schermo apparirà il carattere S di colore nero.

Scriva ora:

```
PRINT PEEK (7910)
```

Comparirà il numero 19, che è appunto il codice numerico, relativo al carattere S (*fig. 24*), contenuto nella locazione di memoria 7910.

ANIMAZIONE E COLORI 2

Eseguiremo ora una seconda versione del programma denominato "*Animazione e colori*", realizzato nella precedente lezione, nel quale verrà ora utilizzata l'istruzione **POKE** per creare l'effetto di illuminazione delle finestre delle case ad ogni passaggio dell'aereo.

Il listato del nuovo programma è riportato nella *fig. 28* e nella *fig. 29*; esaminiamolo limitatamente alle nuove istruzioni che sono state aggiunte.

Come può osservare, le istruzioni 34÷40 richiamano delle subroutine (GOSUB), che iniziano dal numero di linea specificato (2000÷2100).

Ciascuna delle sei subroutine inserisce (POKE) il codice colore 7 (giallo) nelle locazioni di memoria 38778, 38824, 38762, 38808, 38768 e 38837, creando cosí l'illusione che le finestre delle case (la cui posizione è indicata appunto da tali locazioni) si illuminino al passaggio dell'aereo.

La posizione delle finestre illuminate è mostrata nella *fig. 30*.

```
10 REM*ANIMAZIONE E COLORI 2*
15 PRINT"■"
20 GOSUB 1000
21 L1$="                                "
22 POKE36879,24
24 FORSC=24 TO 248 STEP 16
26 FORCO=1 TO 7
27 POKE36879,SC+CO
30 FORX=1 TO 17
34 IFX=2THENGOSUB2000
35 IFX=4THENGOSUB2010
36 IFX=6THENGOSUB2020
37 IFX=8THENGOSUB2030
38 IFX=10THENGOSUB2040
39 IFX=12THENGOSUB2050
40 IFX=17THENGOSUB2100
41 PRINT"■"
42 FORY=1TO5
45 PRINTL1$
47 NEXT Y
48 PRINT"■"
50 PRINTTAB(X)"■ ■"
60 PRINTTAB(X)"■■ ■"
70 PRINTTAB(X)"■■   ■"
80 PRINTTAB(X)"■■ ■"
90 PRINTTAB(X)"■ "
95 FORZ=1TO50:NEXTZ
100 NEXT X
130 NEXTCO
135 POKE36879,SC
136 NEXTSC
138 POKE36879,24
140 PRINT"■":END
1000 PRINT"■■■■■■■■■■"
1010 PRINT"■                ■ ■"
1020 PRINT"■              ■ ■   ■ ■"
1030 PRINT"■    ■  ■    ■ ■ ■ ■   ■  ■ ■"
1040 PRINT"■    ■  ■    ■   ■  ■    ■"
1050 PRINT"■    ■   ■   ■   ■  ■    ■"
1060 PRINT"■    ■    ■  ■    ■  ■    ■"
1070 PRINT"■ ■      ■  ■          ■"
1080 PRINT"■ ■              ■"
1090 PRINT"■ ■              ■"
1095 PRINT"■ ■              ■"
1100 RETURN
```

Fig. 28 – Listato del programma "Animazione e colori 2" per VIC-20.

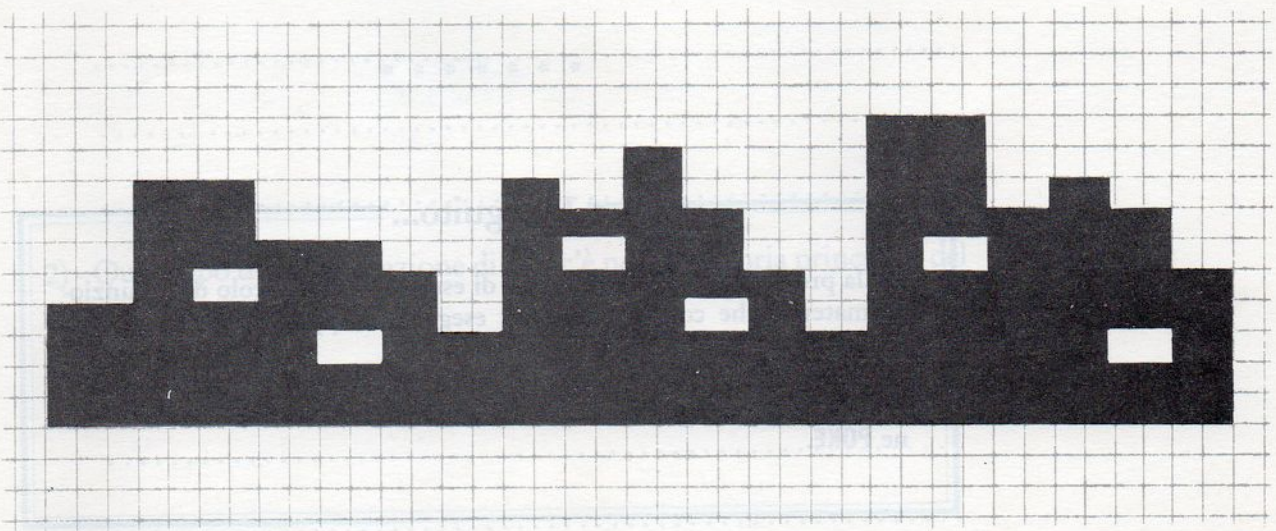
```
2000 POKE38778,7:RETURN
2010 POKE38824,7:RETURN
2020 POKE38762,7:RETURN
2030 POKE38808,7:RETURN
2040 POKE38768,7:RETURN
2050 POKE38837,7:RETURN
2100 POKE38778,0
2110 POKE38824,0
2120 POKE38762,0
2130 POKE38808,0
2140 POKE38768,0
2150 POKE38837,0
2160 RETURN
```

Fig. 29 – Proseguimento del listato del programma "Animazione e colori 2" per VIC-20.

Quando l'aereo giunge all'estremo destro dello schermo, le finestre si "spengono" e ciò viene ottenuto con le istruzioni 21ØØ÷215Ø, ognuna costituita da un'istruzione **POKE** avente la stessa locazione di quelle riportate nelle istruzioni **GOSUB**; in questo caso però il codice colore inserito è Ø, corrispondente al nero, il quale crea appunto l'effetto che la luce delle finestre si spenga al passaggio dell'aereo.

Può altresí rilevare che le istruzioni di diramazione, **IF... THEN...**, previste nelle linee di programma 34÷4Ø, fanno in modo che le finestre si illuminino al passaggio dell'aereo.

Fig. 30 – Posizioni delle "finestre" illuminate, relative al programma della fig. 28 e della fig. 29 (VIC-20).



Infatti, quando la variabile X, che specifica la colonna in cui viene posizionato l'aereo, assume il valore 2, si illumina la finestra relativa alla posizione 38778 della memoria colore (quinta colonna, diciottesima riga); quando l'aereo giunge in corrispondenza della quinta colonna (X=4), si illumina la finestra corrispondente alla posizione 38824 (settima colonna e ventesima riga) e cosí via di seguito sino a quando l'aereo giunge in corrispondenza della diciassettesima colonna per cui si illumina l'ultima finestra, relativa alla locazione 38837 (ventesima colonna e ventesima riga).

Esaminate le modifiche apportate al programma, può ora passare ad eseguirlo tenendo presente quanto precisato nella precedente lezione per quanto concerne le istruzioni relative alla realizzazione dell'aereo (istruzioni 5Ø÷9Ø) e delle case (istruzioni 1Ø1Ø÷1Ø95).

Eseguendo il programma con il comando **RUN**, vedrà appunto illuminarsi sullo schermo ad una ad una le finestre delle case a mano a mano che l'aereo si sposta da sinistra verso destra.

A conclusione della lezione, occorre ancora precisare che, se si compiono errori nell'inserire (POKE) valori in locazioni di memoria errate oppure inserendo valori errati in locazioni corrette, il funzionamento del computer può bloccarsi.

Ciò però non deve assolutamente preoccupare, in quanto è possibile ripristinare le normali condizioni di funzionamento premendo contemporaneamente i tasti RUN | STOP e RESTORE oppure, nei casi piú gravi, spegnendo e riaccendendo, dopo almeno 5÷6 secondi, il computer.

* * * * * * *

In seguito...

Nella prossima lezione avrà modo di esercitarsi nel calcolo delle funzioni matematiche con il computer, eseguendo appositi esercizi di programmazione.

Inoltre, realizzerà altri interessanti effetti di animazione con l'istruzione **POKE**.

CONTROLLO DIDATTICO 8

Domande

1) - Descrivere schematicamente la struttura di un file di dati.

.....

.....

.....

.....

.....

.....

.....

.....

2) - Quale tipo di organizzazione di dati c'è nella memoria principale del computer?

.....

.....

.....

3) - In quale modo sono organizzati: le informazioni registrate su un nastro magnetico; le informazioni registrate su disco; i dati archiviati all'interno di un programma mediante le istruzioni **DATA**?

.....

.....

.....

.....

.....

.....

.....

.....

4) - Disegnare il diagramma di flusso del programma che gestisce un piccolo archivio d'informazione, listato nella *fig. 1* della lezione *Pratica 8*.

5) - Illustrare gli effetti delle seguenti istruzioni:

```
:
100 GET B$
110 IF B$=""THEN 110
:
```

.....

.....

.....

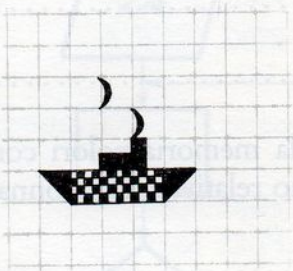
.....

.....

.....

.....

6) - Elaborare un semplice programma che visualizzi la figura schematizzata di una nave la quale, posizionata sulla linea di mezzeria orizzontale dello schermo, si sposti in continuazione da sinistra a destra. La nave deve essere realizzata con i caratteri grafici del computer, come di seguito indicato.



.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

7) - Quali istruzioni occorre dare al computer per fare apparire la lettera A di colore nero nella posizione dello schermo corrispondente alla colonna 15 ed alla riga 18?

.....

.....

.....

8) - Elencare le istruzioni da fornire al computer per fare apparire lo schermo di colore giallo ed il bordo di colore nero.

.....

.....

.....

9) - Quale locazione della memoria colori corrisponde alla posizione (casella) dello schermo relativa alla colonna 7 e alla riga 10?

.....

.....

10) - A quale locazione della memoria caratteri corrisponde la posizione (casella) dello schermo relativa alla colonna 15 e alla riga 8?

.....

.....

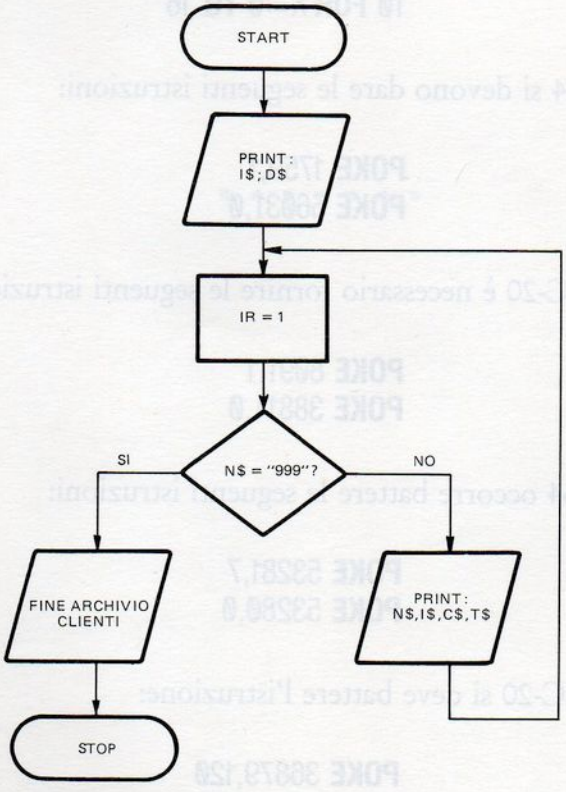
Risposte

1) - Il file è un raggruppamento qualsiasi di informazioni numeriche e stringhe, strutturato e suddiviso in sottogruppi denominati RECORD e CAMPI. Esso è un'organizzazione concreta di dati, poiché viene supportato da nastri e dischi magnetici. Tale organizzazione può essere sequenziale o random (ad accesso casuale).

2) - Nella memoria principale del computer esiste, per la memorizzazione dei dati, solamente l'organizzazione sequenziale.

3) - Le informazioni registrate su nastro magnetico sono organizzate in sequenza (cioè una dopo l'altra), mentre quelle registrate su disco possono essere organizzate in modo sequenziale ed in modo diretto (cioè casuale o random). I dati archiviati nelle istruzioni **DATA** sono organizzati in sequenza.

4) - Il diagramma di flusso relativo al programma che gestisce un piccolo archivio d'informazione ha il seguente aspetto:



5) - Le due istruzioni in oggetto vengono usate per creare pause nel programma ed operare delle scelte. Solamente premendo uno qualsiasi dei tasti della tastiera si può fare riprendere l'elaborazione. Se invece non viene premuto alcun tasto, il programma legge una stringa nulla (rappresentata dai doppi apici) e quindi non può proseguire.

6) - Il programma per la visualizzazione di una nave, relativo al C64, può essere cosí listato:

```
10 FOR X=0 TO 32
20 PRINT"▒▒▒▒▒▒▒▒▒"
25 PRINTTAB(X)"    )    "
30 PRINTTAB(X)"    )    "
35 PRINTTAB(X)"    ▄▄   "
40 PRINTTAB(X)"▼▒▒▒▒▼   "
45 FOR T=1 TO 100:NEXT T
50 NEXT X
55 GOTO 10
60 END
```

Per il VIC-20 occorre modificare l'istruzione 1Ø di tale listato, come di seguito indicato:

```
1Ø FOR X=Ø TO 16
```

7) - Per il C64 si devono dare le seguenti istruzioni:

```
POKE 1759,1
POKE 56Ø31,Ø
```

Per il VIC-20 è necessario fornire le seguenti istruzioni:

```
POKE 8Ø91,1
POKE 38811,Ø
```

8) - Per il C64 occorre battere le seguenti istruzioni:

```
POKE 53281,7
POKE 5328Ø,Ø
```

Per il VIC-20 si deve battere l'istruzione:

```
POKE 36879,12Ø
```

9) - Per il C64:

locazione colore = 55296 + X + 4Ø * Y = 55703

Per il VIC-20:

locazione colore = 384ØØ + X + 22 * Y = 38627

10) - Per il C64:

locazione caratteri = 1Ø24 + X + 4Ø * Y = 1359

Per il VIC-20:

locazione caratteri = 768Ø + X + 22 * Y = 7871

* * * * * * *

PARLA BASIC

Corso teorico-pratico per corrispondenza di

ANALISI E PROGRAMMAZIONE BASIC

Scuola Radio Elettra

10126 Torino - Via Stellone 5

Tel. (011) 674432